Diploma Thesis

Entropy-Preserving Transformation Method

Marcus Hennig

August 14, 2007

Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig und ohne fremde Hilfe angefertigt zu haben. Die verwendete Literatur und sonstige Hilfsmittel sind vollständig angegeben.

Göttingen, 14. August 2007

Contents

1 Introduction

In the past few decades, computer simulations achieved increasing acceptance in natural science as the third major tool linking theory and experiment.

The growing performance of available computer technology has render it possible to examine, relate and characterize large and complex data records from experiments on biomolecular systems. It led to the formulation of models of biomolecular processes, which can be validated and studied utilizing computer simulations. Additionally, computer simulations cannot only simulate biomolecular experiments that can be undertaken in a laboratory, but also experiments whose setup is too sophisticated and costly or that would require time and spacial resolutions that cannot be achieved with existing experimental methods and equipment.

Before the emerging of computer technology the outcome of biomolecular experiments could only be predicted by an approximated or a corse description of the considered system. Analytic solutions existed only for a small number of simple problems. Traditionally, a problem was approached by applying a number of analytical techniques and approximations to find solutions based on physical theories. In comparison, computational techniques are able to tackle more complex system by using numerical methods. For instance, the n-body problem is the problem of finding the dynamics of n bodies as determined by Newton's equation; given the initial positions, masses, and velocities. For more than 2 bodies there is no general analytic solution available but it can be solved numerically. A biomolecule such as lysozyme can have a couple of hundred atoms

and hence biological processes are modeled most promisingly on atomic or molecular level. Addressing questions on the atomic and molecular level by using first principles such as newton dynamics is one of the strengths of computer experiments. Likewise, they have the capability to answer questions why a process occurs by studying possible driving forces.

Numerous biological effects are driven by the free energy $F = U - TS$, composed of the internal energy U, the temperature T, and the entropy S. Minima of the free energy surface correspond to the most probable configurations in phase space. For that reason, decreasing the free energy leads to a more stable configuration.

In the focus of biophysics are mainly proteins surrounded by a solvent such as water. Proteins are biomolecules characterized by a unique sequence of of amino acids (primary structure) and the spatial arrangement of this chain of amino acids (secondary structure) [5]. Solvent entropy is assumed to be the driving force for the arrangement of side chains according to their hydrophilicity. The more the protein exposes hydrophilic side chains to the surrounding solvent molecules the more configurational freedom they have – in other words, solvent entropy increases. In contrast, hydrophobic side chains put constraints on the solvent molecules by forcing an alignment, hence yielding a lower solvent entropy.

Although, the solvent density is analytically known it is impossible to compute the entropy analytically since it requires to determine high dimensional integrals, hence, we rely on numerical methods. Two major problems occur when treating solvents. First, the diffusive motion of the solvent leads to a large configurational space that has to be sampled. Second, the motion of the solvent molecules is governed by a very shallow energy landscape. Hence, the configurational density has a complex topology excluding it from a straightforward analytical estimation.

Tackling the sampling problem was approached by F. Reinhard. He developed a transformation (Permuted Reduction Component Analysis (PRCA)), exploiting the

permutation symmetry of the solvent [18]. Whereas this permutation algorithm provides a promising method to locally condense the configurational density, the topology stays complex. Thus, the transformed configurational density cannot be optimally fitted by Gaussian distribution allowing a simple entropy estimation.

Therefore, we aim at developing a new method to improve Reinhard's permutation reduction by deforming the density such that we can make use of established entropy estimations. With this method we want to contribute to the understanding of biological processes such as protein folding. The goal of this work is to elucidate the problem of solvent densities and to develop a method that lays the ground for solvent entropy calculations and likewise enables to estimate entropies from highly unharmonic system.

2 Theory

2.1 Canonical Ensemble

Classical statistical mechanics is the major tool in molecular dynamics simulations to describe thermodynamic quantities. Though quantum mechanics is the appropriate tool to model on the molecular level it turns out that for many problems classical mechanics gives good results that are in agreement with experiments (reference). Therefore, to introduce the underlying principles we will give a brief elucidation how a many-particle system evolves in time. An example for such a system is a protein surrounded by water molecules. In classical mechanics the time evolution of a system is given by Hamilton's equation

$$\begin{aligned} \dot{x_\alpha} &= \partial\mathcal{H}/\partial p_\alpha, \\ \dot{p_\alpha} &= -\partial\mathcal{H}/\partial x_\alpha, \end{aligned} \tag{2.1}$$

with the Hamiltonian $\mathcal{H}(p, x)$. x_α are the positions and p_α the momenta of all N particles in three dimensions, labeled by the coordinate index $\alpha = 1, \ldots, 3N$.

A typical Hamiltonian of N particles interacting by a potential energy V and kinetic energy K is

$$\mathcal{H}(p,x) = \underbrace{\sum_{\alpha}^{3N} \frac{p_\alpha^2}{2m_\alpha}}_{K(p)} + V(x_1, ..., x_{3N}). \tag{2.2}$$

The $3N$-dimensional space of all possible positions $\mathbb{X} = \{x\}$ is called configuration space. The $3N$-dimensional space of all momenta $\mathbb{P} = \{p\}$ is called momentum space. All possible states of a system are represented by the $6N$-dimensional phase space $\Gamma = (\mathbb{P}, \mathbb{X})$. In a Molecular Dynamics (MD) simulation a sequence of points $(p(t_n), x(t_n))$ in phase space is generated, approximating the thermodynamic behavior of the system.

Treating real systems, like proteins in their environment, requires to take into account that they interact with the environment. To mimic energy exchange a heat bath with a given temperature T is employed. Once the system is in equilibrium with its environment, the average behavior is determined by statistical mechanics. If the number of particles N, the volume V, and the temperature T (NVT-ensemble) is constant, then the probability of finding the system in a state in the vicinity of (p, x) is described by the canonical ensemble [17]

$$\rho(p,x)\, d\Gamma = \frac{1}{Z} \exp\left(-\beta H(p,x)\right) d\Gamma, \tag{2.3}$$

where $\beta = 1/k_B T$, k_B is the Boltzmann constant, T the system temperature and $\mathcal{H}(p,x)$ its Hamiltonian. $d\Gamma \sim dpdx$ is the phase space volume [17].

The normalization factor in equation 2.3, the so-called partition function, is

$$Z = \int_\Gamma \exp\left(-\beta H(p,x)\right) d\Gamma. \tag{2.4}$$

2.2 Free Energy

Three dimensional structures or so-called conformations of proteins are in the focus of structural biology, since they affect their function (reference). Free energy determines the stability of conformational preferences of proteins. To the free energy one considers domains in configurational space within which the protein stays sufficiently long before it explores other parts of configurational space. One refer to these domains as protein conformations. Hence, the set of all possible protein conformations can be described by a family of disjoint finite subsets $\mathbb{X}_i$ of the configuration space $\mathbb{X}$.

As illustrated in figure 2.1 the native state of the protein $\mathbb{X}_N$ and the unfolded state $\mathbb{X}_U$ are described by an ensemble of configurations with a statistical weight given by the Boltzmann factor.

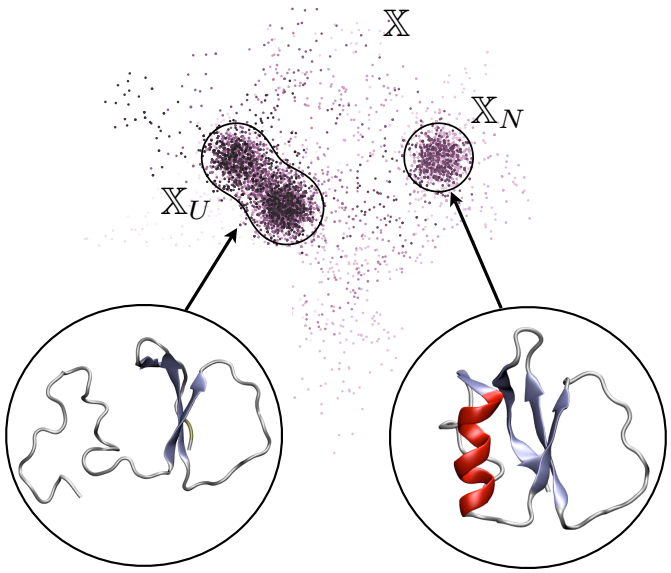


Figure 2.1: Two possible conformations of ?(ask Martin Stumpe) are displayed. The folded conformation corresponds to the subset $\mathbb{X}_N$, and the unfolded to $\mathbb{X}_U$. Each subset is encircled by a contour.

Which of all possible conformations is preferred by the protein can be answered by calculating their probabilities. To start with, we define the partition function of conformation $\mathbb{X}_i$

$$Z_i = \int_{\mathbb{X}_i} \exp\left(-\beta V(x)\right) dx. \tag{2.5}$$

The probability of conformation $\mathbb{X}_i$ is given by

$$P_i = \frac{Z_i}{Z}, \tag{2.6}$$

where Z is the partition function. We introduce the free energy of this conformation

$$F_i = -k_B T \ln Z_i. \tag{2.7}$$

For free energy differences between state i and j we obtain

$$\Delta F_{ij} = F_j - F_i = k_B T \ln \frac{Z_i}{Z_j} \tag{2.8}$$

If $\Delta F_{ij} >> 0$ it is more likely to find protein in conformation $\mathbb{X}_i$, whereas $\Delta F_{ij} = 0$ tells us that both conformations $\mathbb{X}_i$ and $\mathbb{X}_j$ occur with the same probability.

2.3 Total and configurational Entropy

Entropy is in the focus of our work. In this section we want to elaborate important features of entropy. The entropy is defined as expectation value of the logarithm of the probability density ρ (equation 2.3)

$$\mathcal{S}[\rho(p,x)] = -k_B \int_\Gamma \rho\left(p,x\right) \ln \rho\left(p,x\right) d\Gamma, \tag{2.9}$$

where k_B is the Boltzmann constant. $d\Gamma$ is the phase space volume. Often we refer to it as total entropies since it involves the momentum contribution.

Mathematically speaking, the entropy is a functional mapping $\mathcal{S}[.]$ of a probability density function ρ to a real number. For physically meaningful densities [24] the entropy is always positive.

A way to conceive the meaning of entropy, is to consider it as a measure of the volume of phase space that is accessible to the system. Regions of phase space with $\rho = 0$ do not affect the entropy. A large $\mathcal{S}$ implies that a large region of phase space can be accessed by the system, low value signifies that system is constrained. Alternatively, it reflects our ignorance of the exact state of the system.

The nonlinear dependence of $\mathcal{S}[.]$ on ρ does n permit a convenient additivity property. If the density ρ can be written as a product of two other densities functions $\rho(x, y) = \rho(x)\rho(y)$, the entropy split into subspace entropies, hence, it is additive

$$\mathcal{S}\left[\rho(x)\rho(y)\right] = \mathcal{S}[\rho(x)] + \mathcal{S}[\rho(y)] \tag{2.10}$$

The N-body Hamiltonian (equation 2.2) splits into two uncoupled components, namely the kinetic energy K and potential energy V, consequently, the phase space density (equation 2.3) factorizes $\rho(p, x) = \rho(p)\rho(x)$, with

$$\rho(x) = \frac{1}{Z_{\mathbb{X}}} \exp\left(-\beta V(x)\right) \tag{2.11}$$

$$\rho(p) = \frac{1}{Z_{\mathbb{P}}} \exp\left(-\beta K(x)\right) \tag{2.12}$$

$Z_{\mathbb{X}}$ and $Z_{\mathbb{P}}$ are the normalization factors. Total entropy splits into

$$\mathcal{S}[\rho(p, x)] = \mathcal{S}[\rho(x)] + \mathcal{S}[\rho(p)] \tag{2.13}$$

$S[\rho(p)]$ is the momentum entropy and $\mathcal{S}[\rho(x)]$ is the configurational entropy. The momentum entropy can be effortlessly obtained [17]; since it is not relevant for our work, we will concentrate in the following chapters on the configurational density, which cannot not be computed in a straightforward manner.

Experiments have proven that the configurational entropy of the solvent plays a crucial role in protein folding and other processes like the formation of lipid bilayers [5] [11] [6]. Thus, for a protein in a solvent the configurational entropy $\mathcal{S}[\rho(x)]$ may be further slit into the contributions from the solvent and protein by partitioning the configurational space into a protein and solvent subspace: $\mathbb{X} = (\mathbb{X}_\mathrm{S}, \mathbb{X}_\mathrm{P})$. The probability density can be reduced to one subspace by integrating the configurational density over the other subspace, hence

$$\rho(x_S) = \int_{\mathbb{X}_P} \rho(x_S, x_P) dx_P, \tag{2.14}$$

$$\rho(x_P) = \int_{\mathbb{X}_S} \rho(x_S, x_P) dx_S. \tag{2.15}$$

where $x = (x_S, x_P)$ is the configurational vector of the protein-solvent system consisting of x_S the the configurational vector of the the solvent and x_P the configurational vector of the protein. However, splitting entropy into protein entropy $\mathcal{S}[\rho(x_P)]$ and solvent entropy $S[\rho(x_S)]$ – we skip the term configurational entropy, and refer to it simply as entropy – can not be carried out in a physically meaningful way since correlations between both subspaces are neglected. Because $\mathcal{S}[\rho(x, y)] \leq \mathcal{S}[\rho(x)] + \mathcal{S}[\rho(y)]$ [7], a correction term is required

$$\mathcal{S}[\rho(x_S, x_P)] = \mathcal{S}[\rho(x_S)] + \mathcal{S}[\rho(x_P)] + \Delta_\mathrm{corr}. \tag{2.16}$$

Knowledge of the subspace densities $\rho(x_S)$ and $\rho(x_P)$ is not sufficient to determine the entropy of the whole protein-solvent system $\mathcal{S}[\rho(x_S, x_P)]$. Figure 2.3 illustrates that the product of the subspace densities $\rho(x_S)\rho(x_P)$ give rise to a different density as the original density $\rho(x_S, x_P)$. Furthermore, the product of the subspace densities $\rho(x_S)\rho(x_P)$ yields a smaller entropy as the one obtained from the original density $\rho(x_S, x_P)$, since correlations are not accounted. Correlations between the atom constrain the the part of phase space they can access, hence lower the entropy.

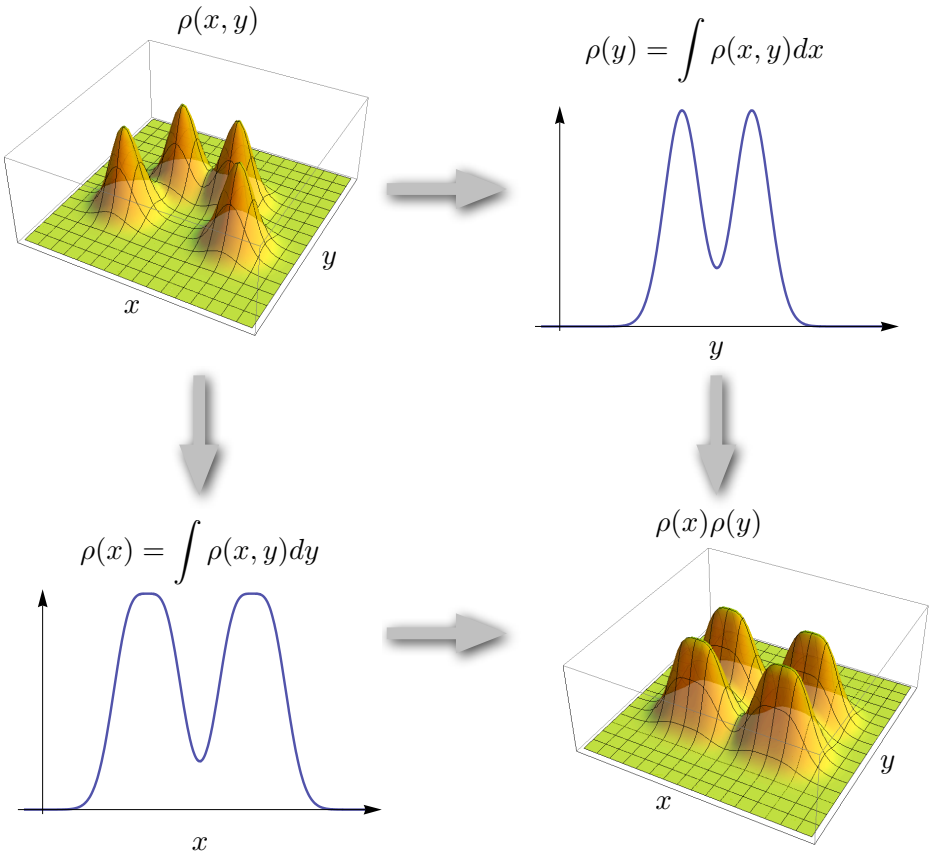


Figure 2.2: For a given joint density $\rho(x, y)$, we can get by integrating over one coordinate the according marginal densities $\rho(x)$ and $\rho(y)$, their product $\rho(x)\rho(y)$ does not in general reconstruct $\rho(x, y)$. Only if x and y are statistically independent, they don't correlate, then $\rho(x, y) = \rho(x)\rho(y)$. A theorem in mathematics states that the sum of the entropy of the marginal densities is always greater than the entropy from the joint density $\rho(x, y)$: $\mathcal{S}[\rho(x)] + \mathcal{S}[\rho(y)] \geq \mathcal{S}[\rho(x, y)]$ [7].

3 Entropy Estimation

3.1 Protein Entropy Calculated from the Covariance Matrix

If we try finding an analytic ansatz for the system consisting of protein and solvent, the obstacle we face, is the different behavior of both. Fluctuating in the vicinity of a single stable, well-defined structure the protein sticks to a small part of its configurational space, lysozyme gives a illustrative example (figure 3.1). In contrast, the motion of solvent molecule is utterly diffusive exploring a large part of configurational space.

In this section we will present a straightforward method to compute the configurational entropy of the protein from the covariance matrix, which can be obtained readily from cartesian coordinates of an ensemble of protein structures. These structures might come from an MD simulation or be generated by a Monte Carlo Simulation (reference).

In the following, for clarity reasons, we skip the notation x_S and use x instead not to confuse with the configurational vector $x = (x_S, x_P)$ of the protein-solvent system. Moreover, with entropy S we refer to the configurational entropy of the protein subspace $S_{\mathbb{X}_S}$.

Karplus developed a method to estimate the entropy S by fitting the density with an analytic ansatz [1] [9]. His ansatz based on the following motivation: Attention centers mostly on the entropy of stable conformations of proteins, like, for example the folded conformation. Fortunately, these conformations occur in the close vicinity

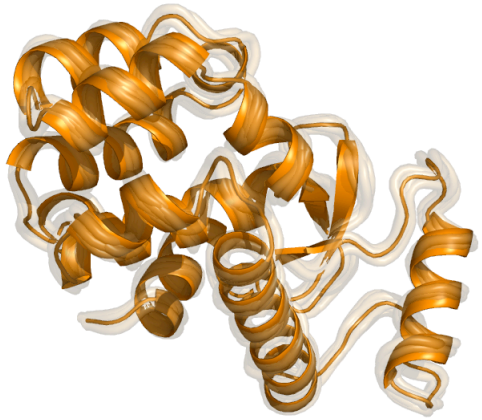


Figure 3.1: Depicted is the trajectory of lysozyme from a MD simulation, to demonstrate the localized configurational space explored by a protein. Plotted is the average structure (orange cartoon plot) and the structure at each time frame (transparent light orange)

of local or even global minima of the free energy surface. Hence, a quasi-harmonic approximation of free energy [9] allows to assume that configurational density of protein can be approximated sufficiently accurate by a Gaussian

$$\rho(x) = \frac{1}{(2\pi)^{3N/2}\sqrt{\det \mathbf{C}}} \exp\left[-\frac{(x-\mu)^T \mathbf{C}^{-1}(x-\mu)}{2}\right] \tag{3.1}$$

$\mathbf{C}$ is the covariance matrix, and μ the midpoint.

For a given ensemble of M protein structures (generated by MD or MC simulations) $\{x^{(m)}\}_{m=1,...,M}$ – where $x = (x_1, ...x_{3N})$ denotes the position of the N atoms of a protein written as a $3N$-dimensional vector. The parameters of the density (equation 3.1) can be estimated simply as follows

$$\mathbf{C}_{\alpha\beta} = \langle x_\alpha x_\beta \rangle - \langle x_\alpha \rangle \langle x_\beta \rangle, \tag{3.2}$$

$$\mu = \langle x \rangle. \tag{3.3}$$

x_α are the coordinates of the atom positions and the angular brackets $\langle . \rangle$ represent the ensemble average, which can be calculated with ease by the time average since we assume ergodicity of our system [15]. Consequently, the ensemble average of a function $f(x)$ is

$$\langle f\,(x) \rangle = \frac{1}{M} \sum_{m=1}^{M} f(x^{(m)}) \tag{3.4}$$

.

Using density approximation (equation 3.1), the configurational entropy (equation **??**) is

$$S = \frac{k_B}{2} \left(3N + \ln \left[(2\pi)^{3N} \det \mathbf{C} \right] \right) \tag{3.5}$$

However, technical problems come up when equation 3.5 is applied to a protein trajectory, since the covariance matrix $\mathbb{C}$ turns out to be practically singular, in other words has eigenvalues approaching zero as it can be seen in figure 3.1. Very small eigenvalues give rise to negative entropies, which are physically not meaningful. However, entropy differences between two states $\mathbb{X}_1$ and $\mathbb{X}_2$ turned out to be estimated accurately [9] [1].

$$\Delta S = S_{\mathbb{X}_1} - S_{\mathbb{X}_1} = k_B \ln \frac{\det \mathbf{C}_{\mathbb{X}_2}}{\det \mathbf{C}_{\mathbb{X}_1}} \tag{3.6}$$

To circumvent the singularity problem of the covariance matrix, Schlitter [19] [20] suggested an ad-hoc approach utilizing the quantum mechanical harmonic oscillator in the quasi-harmonic approximation. The essence of his approach is to employ the solution of the quantum harmonic oscillator to each vibration mode and to obtain an upper limit of the total entropy (configurational entropy plus momentum entropy).

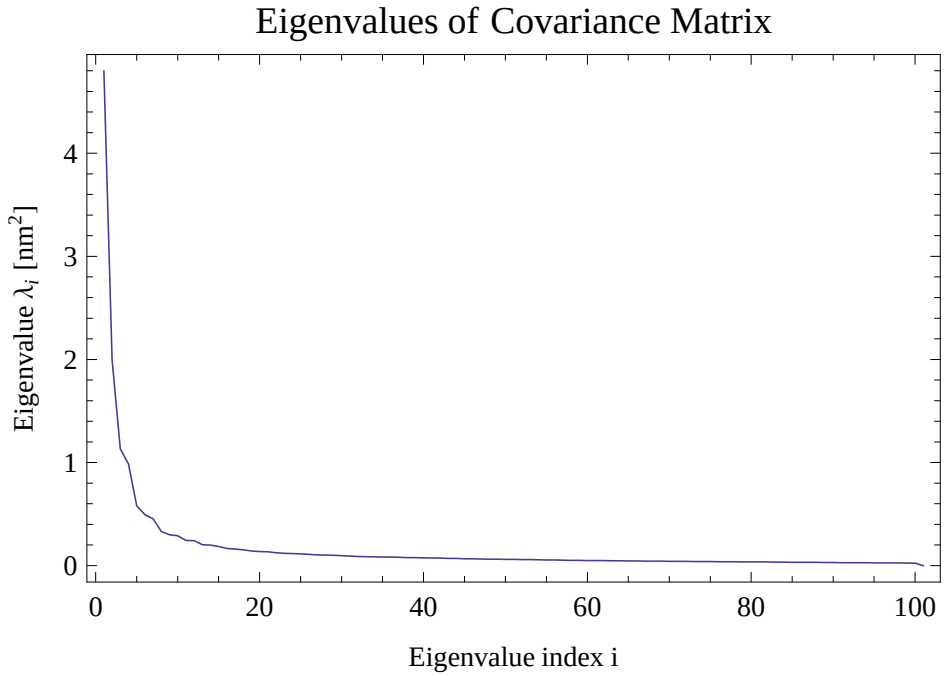


Figure 3.2: First hundred largest eigenvalues of covariance matrix $\mathbf{C}$ from lysozyme trajectory. The covariance matrix is virtually singular ($\det \mathbf{C} \cong 0$)

$$S_{\mathbb{P}_S} + S_{\mathbb{X}_S} \lesssim \frac{k_B}{2} \ln \det \left[\left(1 + \frac{k_B T e^2}{\hbar^2} \tilde{C} \right) \right], \tag{3.7}$$

where $\tilde{\mathbf{C}} = \mathbf{M}^{1/2}\mathbf{C}\mathbf{M}^{1/2}$ is mass-weighted covariance matrix obtained from the covariance matrix $\mathbf{C}$ (equation 3.2) and the mass matrix $M_{\alpha\beta} = m_\alpha \delta_{\alpha\beta}$. m_α is the mass of protein atom with coordinate x_α.

His approach is widely used, however, is should be noted that it is only tailored for proteins. A drawback of Schlitter's method is that it gives only good entropy estimates if the positional fluctuation of the protein obeys at least nearly to a Gaussian distribution. As we will demonstrate later, however, the solvent contribution to the entropy is not accessible to this method.

3.2 Estimation of Solvent Entropies via Permutation Reduction

While several methods exists to compute the protein entropy, it is challenging to determine the solvent entropy, since the solvent density may not be approximated by a Gaussian distribution. Figure 3.3 illustrates a simple hard sphere example, which cannot be described by a Gaussian distribution. Hence, Schlitter's straightforward approach cannot be applied.

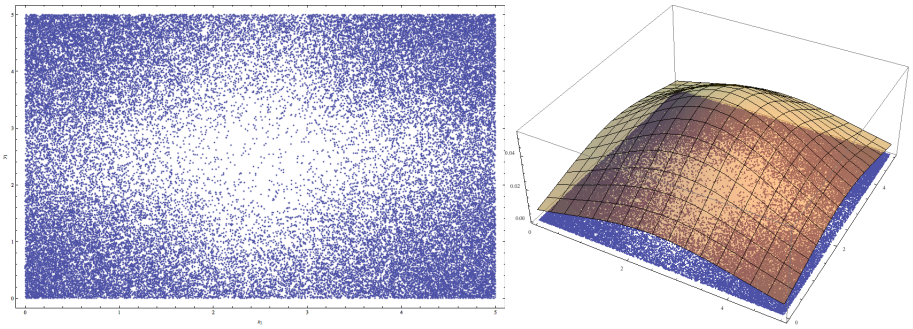
A completely novel and promising approach to compute solvent entropies was developed by F. Reinhard [18]. In this section we will outline his idea how to estimate solvent entropies via permutation reduction from a given trajectory. His approach was motivated by two major problems associated with solvent entropies:

First, it is infeasible to find an analytic ansatz to describe the solvent density, since it exhibits a too complex analytic structure. Due to the repulsive potential for overlapping solvent parts of the configurational space are inaccessible, hence creating holes in the density distribution (figure 3.3). Even larger holes come from the interaction with the protein.

Second, the configurational space explored by the diffusive motion of the solvent molecules is too large to be sampled within a reasonable time by current simulation techniques.

Reinhard's approach tackles the second problem and significantly contributes to the solution of the first one. In a brief description we want to provide the mathematical background of permutation reduction — a term we will use in the following to refer to Reinhard's method. Permutation reduction based is on the idea to exploit the permutation symmetry of the solvent interaction potential and neglects the coupling between protein and solvent.

To start with, in many cases we can assume that the solvent interaction potential is



(a) Configurational density (b) Gaussian estimate

Figure 3.3: Two dimensional projection (on first two eigenvectors of covariance matrix of sample points) of the configurational density of a system consisting of three two-dimensional hard spheres trapped in a box. (a) The repulsive interaction of the spheres will yield parts of configurational space that are not accessible (holes). (b) A gaussian estimate of the density will fail since it not accounts for the hole in the middle.

described by the term

$$V(x) = \sum_{i<j} V_{\mathrm{ss}}(x_i, x_j) \tag{3.8}$$

Where $x = (x_1, ..., x_N)$ denotes the configurational vector of all N solvent molecules. Each single solvent molecule labeled by i may described by $3m$ dimensional vector $x_i = (x_{i,1}, ..., x_{i,m})$ consisting of the three-dimensional positions of its m atoms. Accordingly, the configurational space $\mathbb{X}$ is $3mN$-dimensional. The mutual interaction of solvent molecules is described by a symmetric potential function $V_{\mathrm{ss}}(x_i, x_j)$ with $V_{\mathrm{ss}}(x_i, x_j) = V_{\mathrm{ss}}(x_j, x_i)$.

Obviously, the solvent molecules are indistinguishable, and consequently, we can interchange the (i.e. relabel them) them without changing the value of the total potential (equation 3.8). With this in mind, we introduce a permutation operator on solvent

molecules

$$\mathcal{P}_\pi x = x_\pi = \left(x_{\pi(1)}, ..., x_{\pi(N)}\right) \tag{3.9}$$

π is a permutation of the numbers $\{1, ..., N\}$. These permutations form a finite group denoted by Π with $N!$ elements.

Since the solvent molecules are interchangeable, the potential (equation 3.8) is permutational invariant. And hence, the density (equation 2.11) posses the same invariance

$$\rho(x) \quad = \quad \rho(x_\pi) \quad \forall \pi \in \Pi, \tag{3.10}$$

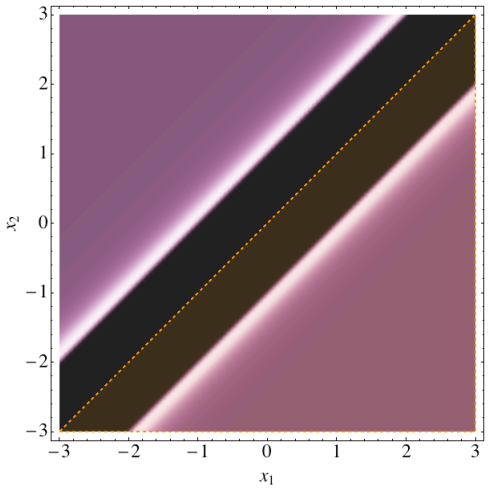


Figure 3.4: Configurational density of a two particle system, whose interaction is described by a Lennard-Jones potential. Due two the permutation symmetry of the potential, the configurational space can be decomposed into 2! subspaces. Regions with vanishing density occur a little bit blurred at the surface of hyper-plane. If there is additionally a protein-solvent interaction, holes will appear within the subspace.

in other words, the solvent configurations x_π and x occur equally likely. As a result, we can relabel solvent molecules without changing thermodynamic quantities as entropy or free energy.

A powerful feature of symmetries is to split the considered space in subsets with equal properties. Here, the permutation symmetry, allows us to split the configurational space $\mathbb{X}$ into $N!$ slices, each having the same thermodynamic properties. We can generate these slices by the following procedure

$$\begin{aligned} \xi &\in \mathbb{X} \\ \mathbb{X}(\xi) &= \{x \in \mathbb{X}: \quad \|x_\pi - \xi\| > \|x - \xi\| \quad \forall \pi \in \Pi \setminus \{1\}\} \\ \mathbb{X}_\pi &= \mathbb{X}(\xi_\pi) \quad \forall \pi \in \Pi \\ \mathbb{X} &= \bigcup_{\pi \in \Pi} \mathbb{X}_\pi \end{aligned} \tag{3.11}$$

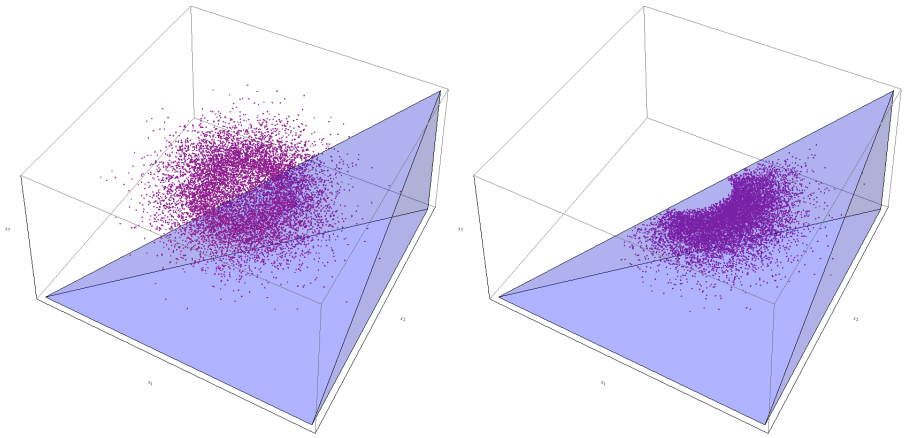
In the above construction the configuration ξ can be chosen arbitrarily (Appendix). It serves as a generator to define the slice $\mathbb{X}(\xi)$, an arbitrary representative of the decomposition, from which we produce $N!$ slices Γ_π by applying all permutations $\pi \in \Pi$, splitting the configurational space $\mathbb{X}$ into $N!$ slices $\mathbb{X}_\pi$.

Let us consider an illustrative example of two one-dimensional particles interacting by an Lennard-Jones Potential. In figure 3.4, we have plotted the resulting configurational density. The configurational space is split into 2 slices due to the permutation symmetry of the density. For one slice we have denoted the boundary by a dashed orange line.

A consequence of equation 3.11 is that for each $x \in \mathbb{X}$ there is a permutation π such that $x \in \mathbb{X}_\pi$, likewise, and more important, for a fixed slice $\mathbb{X}_\sigma$ we can always find a permutation π such that $x_\pi \in \mathbb{X}_\sigma$.

In other words, exploiting the permutation symmetry, for a given configuration ξ we can compress an ensemble of solvent configurations into a small slice $\mathbb{X}(\xi)$ of its configu-

rational space $\mathbb{X}$ without changing any thermodynamic quantity. The remaining space configurational space does not provide any additional thermodynamic information.



(a) Before relabeling (b) After relabeling

Figure 3.5: (a) Ensemble of configurations in a 3-dimensional space before relabeling. (b) After relabeling the ensemble is compressed into 1/3! fraction of the space (blue)

Accordingly, Reinhard suggested to transform a solvent trajectory via permuting (by relabeling each frame) into one slice, we will also refer to as reduced space, such that it renders the trajectory more compact and improves significantly the sampling problem.

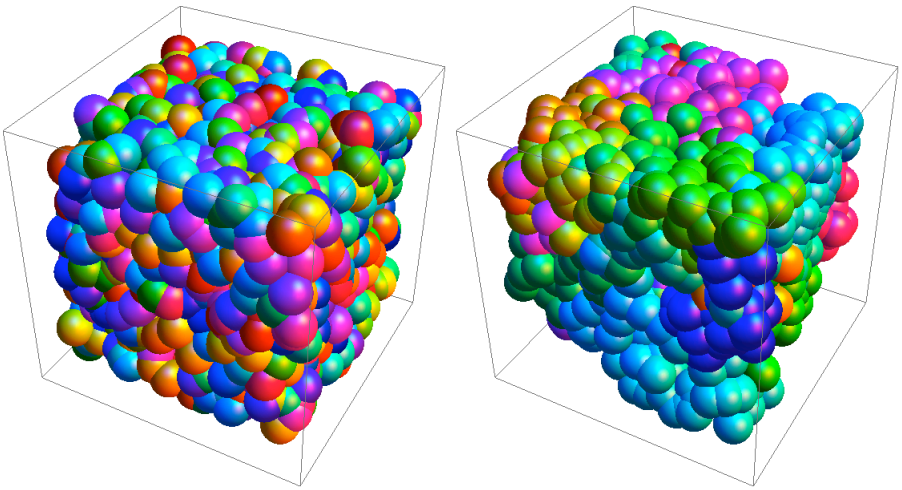
Reinhard found that permutation reduction corresponds to a linear assignment problem, for which many elegant solutions exist. His algorithm works as follows: Find a permutation π of the solvent molecules for each frame bringing the solvent close to a chosen reference position ξ. In mathematical terms, find a π such that+

$$\|x_\pi - \xi\| < \|x_\sigma - \xi\| \quad \forall \sigma \in \Pi \tag{3.12}$$

After relabeling the trajectory is compressed into the reduced space $\mathbb{X}(\xi)$ depending only on the chosen reference configuration ξ

As illustration of Reinhard's permutation reduction we have applied his method to a

trajectory of a simulation of an argon gas trapped in box. The left picture of figure 3.6 shows the diffusive motion of argon atoms in box before relabeling. Each argon atom is colored differently. Different spheres of the same color depict different locations of the atom during diffusion. The right picture shows the atoms after relabeling. Relabeling can be considered as changing the color of the atoms. Positions in the vicinity of a reference atom are assigned to atoms which are close, thus changing their color. As a result the motions of the relabeled atoms is restricted to a small part of the simulation box.



(a) Before relabeling (b) After relabeling

Figure 3.6: Simulation of gas consisting of 216 argon atoms trapped in box (a) Diffussive motion of argon atoms, each colored differently, before relabeling. (b) After relabeling the argon atoms stay close to a reference position, hence, the colors cluster.

After relabeling the ensemble covers only a small fraction of configurational space. The probability density of finding a configuration x in the reduced space $\mathbb{X}(\xi)$ is given by

the conditional density

$$\rho_0(x) = \rho\left(x \mid x \in \mathbb{X}\left(\xi\right)\right) = \begin{cases} N!\rho(x) & x \in \mathbb{X}\left(\xi\right) \\ 0 & \text{otherwise} \end{cases}. \tag{3.13}$$

Hence, ρ (equation 2.11) can be easily recovered by applying all possible permutations

$$\rho(x) = \frac{1}{N!} \sum_{\pi \in \Pi} \rho_0\left(x_\pi\right) \tag{3.14}$$

Using the density of the relabeled ensemble (equation 3.13) the total configurational entropy $S[\rho(x)]$ (equation 2.11) is given by

$$S[\rho(x)] = S[\rho_0(x)] + \ln N!. \tag{3.15}$$

Reinhard's permutation reduction improves drastically the sampling problem, further entropy estimation can be applied without the burden of exhaustive sampling, since the density is compressed into a small slice of configurational space. Additionally, the geometric form of the slice is known, what might be beneficial for further analysis.

4 Entropy Preserving Transformation Method

4.1 Holes - The Density-Fit Problem

The permutation reduction of Reinhard led to a representation formula (equation 3.14) of the density ρ as the sum of permutations of a single relabeled density ρ_0 (equation 3.13), which is nonzero on a small part of configurational space — the so-called reduced space. From this the density ρ_0 may be estimated by a Gaussian. However, Reinhard found out that estimating ρ_0 with a Gaussian is not the best choice to produce accurate entropy estimates of a relabeled density. The reason is that the relabeled density — though it is more localized than the original one — is still too anharmonic to be fitted by a Gaussian.

In this section we will elucidate that holes, which are parts of configurational space with vanishing density, cause anharmonicity. These holes are not eliminated by Reinhard's method. Therefore, we suggest in the adjacent chapters a method, that deforms the density such that the holes disappear and the entropy is preserved.

By means of a simple protein-solvent model we will show that holes render an easy analytic ansatz infeasible and require another approach to determine the entropy. To model the protein-solvent system, we will assume that the protein is fixed in configurational space, motivated by the experience that proteins often assume a stable con-

formation (reference). The Protein consists of N_P atom at positions $y_j,\ j = 1, ..., N_P$. Whereas the solvent is assumed to move diffusively around the protein in the simulation box, in which both are located. The positions of the N_S solvent molecules are $x_i,\ i = 1, ..., N_S$. The potential of the system is assumed to have the form

$$V(x) = \sum_{i>j} V_{\mathrm{ss}}(x_i, x_j) + \sum_{i,p} V_{\mathrm{sp}}(x_i, y_p) + V_{\mathrm{pp}}, \tag{4.1}$$

where V_{ss} describes the symmetric interaction potential between the solvent molecules. The interaction of the solvent with the protein is modeled with a potential V_{sp}. V_{pp} stands for the potential energy of the protein. Details of the potential functions are not important for further consideration. To explain the coarse structure of the resulting density, we assume that the interaction exhibit a repulsive character, because no two atoms can be at the same place (Pauli principle), hence,

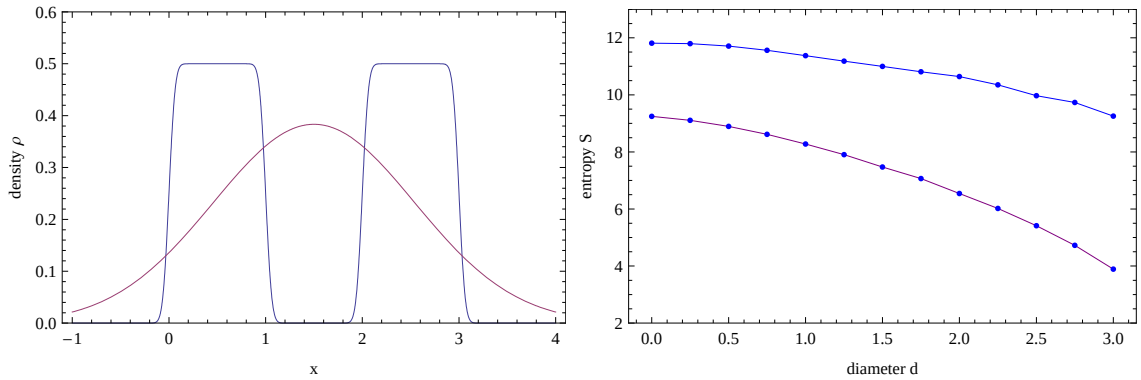
$$\begin{aligned} \lim_{x_i \to x_k} V_{\mathrm{ss}}(x_i, x_k) &= \infty, \\ \lim_{x_i \to y_j} V_{\mathrm{sp}}(x_i, y_j) &= \infty. \end{aligned} \tag{4.2}$$

The repulsive character of the potentials tells us, that the total potential V (equation 4.1) tends to ∞, in other words the density (equation 2.11) vanishes, if either two solvent molecules get converge or a solvent molecules gets the volume occupied by the protein. Mathematically speaking

$$\forall x \in \mathbb{X}_S \text{ with } i \neq k, x_i = x_k : \quad \rho(x) = 0, \tag{4.3}$$

$$\forall y_p \in \mathbb{X}_P\ \forall x \in \mathbb{X}_S \text{ with } x_i = y_j : \quad \rho(x) = 0. \tag{4.4}$$

Hence, the two-particle-interactions V_{ss} of N_S solvent molecules give rise to $N_S(N_S - 1)$ holes. Likewise, the solvent-protein V_{sp} interaction leads to inaccessible parts of configurational space. Obviously, the resulting holes in the configurational density ρ will



(a) Holes (b) 3x2D Hard Spheres

Figure 4.1: (a) Sketch of a Gaussian estimate (purple line) of a density (blue line) that has a hole in the middle and vanishes at its boundary. The Gaussian estimate is maximal at the hole in the center, where the actual density vanishes. (b) Simple hard sphere system to exemplify solvent entropy. Three two-dimensional hard spheres of diameter d moving in a 2D square box with length $a = 5$, a protein was modeled by a hard sphere of diameter $d_p = 3$ located in the center of the box. We generated several ensembles with diameters of the solvent disk ranging from 0 to 3. All ensembles were relabeled according to Reinhard's reduced permutation and the entropy was estimated (blue) by fitting a Gaussian to the ensemble density. The real entropy was computed with a Monte-Carlo Method (purple line).

still be present in the relabeled density ρ_0 for symmetry reasons (equation 3.10). Consequently, the density ρ_0 is still not shaped in a way that would allow to approximate it using a Gaussian since holes would be neglected (figure 4.1 (a)).

It can be proven (Appendix) that the repulsive interactions between solvent molecules give rise to holes in the relabeled density ρ_0 at the surface of the reduced space and that holes resulting from interaction with the protein can occur in the interior of the reduced space.

For a simple system consisting of three hard spheres moving in a box and a protein, modeled by a hard sphere in the center the (relabeled) density posses holes due to the repulsive potential the of disks and the area excluded by the "protein". These holes render the relabeled density anharmonic, which means that it cannot be fit by a Gaussian. Hence, estimating the entropy by fitting the relabeled density with a Gaussian (blue curve in figure 4.1 (b)) leads to overestimation of the real entropy (purple line in figure 4.1 (b)).

4.2 Entropy Preserving Transformations

4.2.1 Analytical Representation by Incompressible Flow

We demonstrated that the relabeled density (equation 3.13) may be non-gaussian. Hence, it renders a analytic fit ansatz, for example a Gaussian fit, infeasible. Therefore, we will try to find "deforming transformations" $\mathbf{f} : \mathbb{X}_S \to \mathbb{X}_S$ in the configurational space of the solvent that warp the density into a more Gaussian one without changing the entropy of the considered system. To this end, it is desirable to "close holes".

If we apply an arbitrary smooth and invertible transformation $\mathbf{f}$ to the configurational vector $\mathbf{x}$ of the solvent — mathematically, a random variable — the corresponding transformation of the density (equation 3.13) is (reference)

$$\mathbf{f}[\rho_0](\mathbf{y}) \equiv \rho_0(\mathbf{x}) \det\left[\mathbf{J}_\mathbf{f}(\mathbf{x})\right]^{-1}, \text{ with } \mathbf{x} = \mathbf{f}^{-1}(\mathbf{y}), \tag{4.5}$$

where $\mathbf{f}[\rho_0]$ denotes the transformed density and $\mathbf{J}_\mathbf{f}$ is the Jacobian matrix of the transformation $\mathbf{f}$. The transformed density $\mathbf{f}[\rho_0]$ has the configurational entropy

$$\mathcal{S}\left[\mathbf{f}[\rho_0]\right] = -k_B \int \mathbf{f}[\rho_0](\mathbf{y}) \ln \mathbf{f}[\rho_0](\mathbf{y}) d\mathbf{y}, \tag{4.6}$$

changing variables $\mathbf{y} = \mathbf{f}(\mathbf{x})$ with $d\mathbf{y} = \det \mathbf{J}_\mathbf{f}(\mathbf{x}) d\mathbf{x}$ and using equation 4.5 yields the

entropy transformation

$$\mathcal{S}\left[\mathbf{f}[\rho_0]\right] = -k_B \int \rho_0(\mathbf{x}) \ln\left(\rho_0(\mathbf{x}) \det\left[\mathbf{J}_\mathbf{f}(\mathbf{x})\right]^{-1}\right) d\mathbf{x} \tag{4.7}$$

$$= \mathcal{S}\left[\rho_0\right] + \underbrace{\int \rho_0(\mathbf{x}) \ln\left(\det\left[\mathbf{J}_\mathbf{f}(\mathbf{x})\right]\right) d\mathbf{x}}_{\text{entropy change}}. \tag{4.8}$$

Transformations $\mathbf{f}$ that compress ($\det\left[\mathbf{J}_\mathbf{f}(\mathbf{x})\right] > 1$) or expand ($\det\left[\mathbf{J}_\mathbf{f}(\mathbf{x})\right] < 1$) the density ρ_0 are too complicated to handle, since it would be computationally expensive to correct the entropy change. Equation 4.8 indicates that transformations with unit Jacobian are entropy-preserving, since the logarithm of the Jacobian determinate vanishes. Hence,

$$\det\left[\mathbf{J}_\mathbf{f}(\mathbf{x})\right] = 1 \Rightarrow S\left[\mathbf{f}[\rho_0]\right] = S\left[\rho_0\right]. \tag{4.9}$$

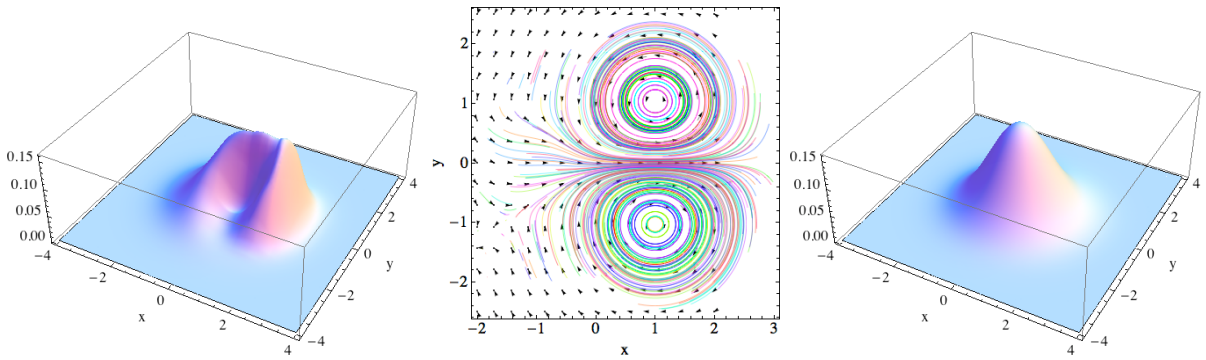
Consequently, we focus on transformations f with unit Jacobian as possible candidates for deforming the density in such a way that the entropy is not preserved. These class of transformations is referred to as volume-preserving maps. In the following we use the terms volume- and entropy preserving transformations (maps) interchangeably.

A construction of smooth volume-preserving maps is provided by the "density theorem" [2]. According to this theorem smooth volume-preserving maps can be generated from solutions of a first order ordinary differential equation (ODE)

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}\left(\mathbf{x}\right), \tag{4.10}$$

describing the motion of a "particle" in an incompressible flow v. An incompressible flow is described by a field in which the divergence of the velocity is zero,

$$\nabla \cdot \mathbf{v} = \sum_i \frac{\partial v_i}{\partial x_i} = 0. \tag{4.11}$$



(a) Initial density (b) Divergence-free field (c) Reshaped density

Figure 4.2: An initially non-gaussian density (a) is deformed by a velocity field (b) such that the result (c) is more gaussian shaped. Both densities have the same entropy as the deformation takes place in an incompressible flow.

We introduce the continuous transformation $\mathbf{f_v}(t, \mathbf{x})$ denoting the solution of the the ODE (equation 4.10) at t with $\mathbf{x}$ as initial condition at $t = 0$. We show that the determinant of the Jacobian of $\mathbf{f_v}(t, .)$ is equal to one. As shown in appendix 8.2 the following identity holds

$$\partial_t \ln \det \left(\frac{\partial \mathbf{f_v}(t, \mathbf{x})}{\partial \mathbf{x}} \right) = (\nabla \cdot \mathbf{v}) \left(\mathbf{f_v}(t, \mathbf{x}) \right). \tag{4.12}$$

The zero-divergence of v (equation 4.11) implies that the right side of equation 4.12 is zero, hence, the determinant of the Jacobian is time-independent. Employing the initial condition $f_\mathbf{v}(0, x) = x$ we obtain

$$\det \left(\frac{\partial \mathbf{f_v}(t, \mathbf{x})}{\partial \mathbf{x}} \right) = \det \left(\frac{\partial \mathbf{x}}{\partial \mathbf{x}} \right) = 1, \tag{4.13}$$

hence, $\mathbf{f_v}(t, .)$ is a entropy-preserving (implication 4.9).

Furthermore, equation 4.10 describes the characteristic curve [21] of an advection equation, a partial differential equation describing the propagation of the density ρ in

the velocity field $\mathbf{v}$.

$$\frac{\partial \rho}{\partial t} + \mathbf{v} \cdot \nabla \rho = 0 \tag{4.14}$$

As a result, given the divergence-free field v and flow time $t > 0$, we can construct an entropy-preserving transformation $\mathbf{f}_{\mathbf{v}}(t, \mathbf{x})$ by moving the configurational vector $\mathbf{x}$ for a certain time t along the streamline of an incompressible — divergence-free — velocity field. However, this construction does not work satisfactorily, since we do not have a tool to build every possible smooth divergence-free velocity field to generate all entropy-preserving maps

$$\mathbb{G} = \{\mathbf{f}_{\mathbf{v}}(t, .) \mid \quad \nabla \cdot \mathbf{v} = 0\}, \tag{4.15}$$

which form a group, meaning that the composition of two arbitrary elements of $\mathbb{G}$ is also entropy-preserving, hence a member of the set $\mathbb{G}$. Later, we will exploit the group property to decompose a entropy-preserving transformation into a composition of low-dimensional transformations, since it is infeasible to find a high-dimensional transformation, warping the density into a more Gaussian one.

4.3 Divergence Free Wavelets

The construction of entropy-preserving transformations depends strongly on divergence free fields. Therefore, we will explain in this section how to construct a basis of divergence-free vector fields. In the following we concentrate merely on mathematical aspects and therefore, instead of speaking of a particular configurational space we will switch to the n-dimensional euclidian space $\mathbb{R}^n$ but keeping in mind that we are seeking for a divergence-free vector field in the configurational space of the solvent $\mathbb{X}_S$.

The linear vector space (reference) of divergence-free vector fields is

$$\mathbb{H}_{\mathrm{div},0}\left(\mathbb{R}^n\right) = \left\{\mathbf{u} \in \left(\mathbb{L}^2\left(\mathbb{R}^n\right)\right)^n \mid \quad \nabla \cdot \mathbf{u} = 0\right\}, \tag{4.16}$$

where $\left(\mathbb{L}^2\left(\mathbb{R}^n\right)\right)^n$ indicates the space of square integrable n-dimensional vector functions, for which $\int \|\mathbf{u}\left(\mathbf{x}\right)\| d^n\mathbf{x} < \infty$ holds. The linearity of the function space $\mathbb{H}_{\mathrm{div},0}$ allows us to write every divergence-free field as a simple linear combination of this basis. Consequently, we can easily parametrize the group of entropy-preserving transformations (equation 4.15).

P.G. Lemarie-Rieusset designed a compactly supported and divergence-free wavelet base for $\mathbb{H}_{\mathrm{div},0}\left(\mathbb{R}^n\right)$ by an algebraic construction based on tensor products and biorthogonal Multiresolution analyses. Wavelets are functions that allow to expand arbitrary functions into integer scaled and translated copies of a single waveform, the so-called (mother-) wavelet [13].

In recent years divergence-free wavelets became a popular tool in hydrodynamics. They are used to analyze two-dimensional flows and as well to compute the Navier-Stokes solution for the driven cavity problem [22]. We propose to use divergence-free wavelets to construct smooth entropy-preserving transformations.

In the following we will provide the basics of the construction principle originally developed by Lemarie-Rieusset and Urban [23] [3] [4]. We will focus on the implementation of two-dimensional divergence-free vector wavelets base since we will use them later.

4.3.1 Multiresolution Analysis (MRA)

To begin with, we briefly review the concept of multiresolution analyses, that are necessary to construct a wavelets base of a functions space. Multiresolution analyses (MRA) are function approximation spaces, in the one-dimensional case, defined by a sequence of closed subspaces $\left(V_j\right)_{j\in\mathbb{Z}}$ fulfilling the following conditions

$$\forall j, \quad V_j \subset V_{j+1}, \tag{4.17}$$

$$\bigcap_{j\in\mathbb{Z}} V_j = \{0\}, \quad \overline{\bigcup_{j\in\mathbb{Z}} V_j} = \mathbb{L}^2(\mathbb{R}), \tag{4.18}$$

$$f \in V_j \Leftrightarrow f(2.) \in V_{j+1}, \tag{4.19}$$

$$\text{There exists a function } \phi \in V_0 \text{ such that } V_0 = \text{span}\{\phi(.-k), \quad k \in \mathbb{Z}\}, \tag{4.20}$$

where ϕ is called scaling function of the MRA. The index j can be understood as a refinement level. From condition 4.19 and 4.20 we can deduce that

$$V_j = \text{span}\{2^{j/2}\phi\left(2^j. - k\right), \quad k \in \mathbb{Z}\}. \tag{4.21}$$

Equation 4.20 means that every function of V_j can be written as a linear combination of integer shifted scaling functions ϕ. Wavelets appear as the relative complement space W_j of V_j in V_{j+1}

$$V_{j+1} = V_j \oplus W_j \tag{4.22}$$

where $\oplus$ is a direct sum. However, equation 4.22 not necessarily mean that both summands are orthogonal. It is possible to find a wavelet function ψ such that $W_0 = \text{span}\{\psi(.-k), \quad k \in \mathbb{Z}\}$. Likewise, we can deduce that $W_j = \text{span}\{\psi_{j,k}, \quad k \in \mathbb{Z}\}$, where $\psi_{j,k} = 2^{j/2}\psi(2^j. - k)$. A repeated decomposition of V_j yields the wavelet decomposition of the function space

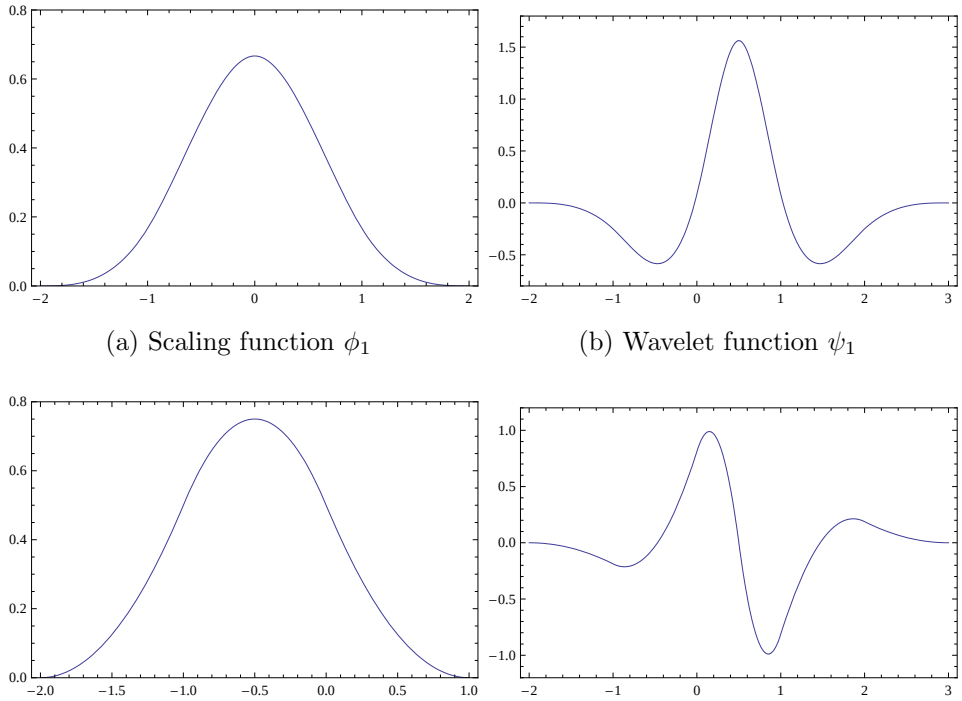
$$\mathbb{L}^2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} W_j. \tag{4.23}$$

Consequently, every function f in $\mathbb{L}^2(\mathbb{R})$ can be written as

$$f(x) = \sum_{j\in\mathbb{Z}} \sum_{k\in\mathbb{Z}} c_{j,k}\psi_{j,k}(x) \tag{4.24}$$

4.3.2 Multivariate MRA

The above consideration can be extended to multi-dimensions. A way to obtain multivariate wavelets is to use products of one-dimensional functions. For the two-dimensional case we elucidate the construction of two-dimensional wavelets from two different one-dimensional MRAs $\left(V_j^0\right)_{j\in\mathbb{Z}}$ and $\left(V_j^1\right)_{j\in\mathbb{Z}}$.



(a) Scaling function ϕ_1 (b) Wavelet function ψ_1

(c) Scaling function ϕ_0 (d) Wavelet function ψ_0

Figure 4.3: Scaling and wavelet functions of two MRAs $\left(V_j^0\right)$ and $\left(V_j^1\right)$ related by $\phi_1(x)' = \phi_0(x) - \phi_0(x-1)$ and $\psi_1(x)' = 4\psi_0(x)$.

We start with constructing an MRA of $\left(\mathbb{L}^2\left(\mathbb{R}^2\right)\right)^2$ from two different MRAs $\left(V_j^0\right)_{j\in\mathbb{Z}}$ and $\left(V_j^1\right)_{j\in\mathbb{Z}}$ — with scaling and wavelet functions (ϕ_0, ψ_0) and (ϕ_1, ψ_1) respectively —

by

$$\left[\mathbf{V}_j = \left(V_j^1 \otimes V_j^0\right) \times \left(V_j^0 \otimes V_j^1\right)\right]_{j\in\mathbb{Z}} \tag{4.25}$$

Due to the function space $\mathbf{V}_j$ is spanned by two vector-valued scaling functions

$$\boldsymbol{\phi}_{(1,0)}(x,y) = \phi_1(x)\phi_0(y)\begin{pmatrix}1\\0\end{pmatrix}, \quad \boldsymbol{\phi}_{(0,1)}(x,y) = \phi_0(x)\phi_1(y)\begin{pmatrix}0\\1\end{pmatrix} \tag{4.26}$$

Hence, $\mathbf{V}_j = \text{span}\{\boldsymbol{\phi}_{(1,0),j,\mathbf{k}},\ \mathbf{k}\in\mathbb{Z}^2\} \oplus \text{span}\{\boldsymbol{\phi}_{(0,1),j,\mathbf{k}},\ \mathbf{k}\in\mathbb{Z}^2\}$, where $\boldsymbol{\phi}_{e,j,\mathbf{k}}(x,y) = 2^j\boldsymbol{\phi}_e\left(2^jx - k_x, 2^jy - k_y\right)$ with $e\in\{(1,0),(0,1)\}$. Employing the decomposition of both one-dimensional MRAs V_j^0 and V_j^1 (equation 4.22) the complement of $\mathbf{V}_j$ in $\mathbf{V}_{j+1}$ — the wavelet space $\mathbf{W}_j$ — becomes a direct sum of tensor spaces consisting of combinations of V_j^0, W_j^0 and V_j^1, W_j^1

$$\left[\mathbf{W}_j = \begin{pmatrix}V_j^1\otimes W_j^0\\ V_j^0\otimes W_j^1\end{pmatrix} \oplus \begin{pmatrix}W_j^1\otimes V_j^0\\ V_j^0\otimes W_j^1\end{pmatrix} \oplus \begin{pmatrix}W_j^1\otimes W_j^0\\ W_j^0\otimes W_j^1\end{pmatrix}\right]_{j\in\mathbb{Z}}. \tag{4.27}$$

The wavelet space $\mathbf{W}_0$ is spanned by six wavelets functions which are obtained by building tensor products of the appropriate functions

$$\begin{aligned}
&\boldsymbol{\psi}_{(1,0)}{}^{(0,1)}(x,y) = \phi_1(x)\psi_0(y)\begin{pmatrix}1\\0\end{pmatrix}, \quad \boldsymbol{\psi}_{(0,1)}{}^{(0,1)}(x,y) = \phi_0(x)\psi_1(y)\begin{pmatrix}0\\1\end{pmatrix},\\
&\boldsymbol{\psi}_{(1,0)}{}^{(1,0)}(x,y) = \psi_1(x)\phi_0(y)\begin{pmatrix}1\\0\end{pmatrix}, \quad \boldsymbol{\psi}_{(0,1)}{}^{(1,0)}(x,y) = \psi_0(x)\phi_1(y)\begin{pmatrix}0\\1\end{pmatrix},\\
&\boldsymbol{\psi}_{(1,0)}{}^{(1,1)}(x,y) = \psi_1(x)\psi_0(y)\begin{pmatrix}1\\0\end{pmatrix}, \quad \boldsymbol{\psi}_{(0,1)}{}^{(1,1)}(x,y) = \psi_0(x)\psi_1(y)\begin{pmatrix}0\\1\end{pmatrix}.
\end{aligned} \tag{4.28}$$

By integer scaling and shifting we obtain the family of wavelets

$$\left\{\boldsymbol{\psi}^{\epsilon}_{i,j,\mathbf{k}}(\mathbf{x}) = 2^j\boldsymbol{\psi}^{\epsilon}_i\left(2^j\mathbf{x} - \mathbf{k}\right)\right\} \tag{4.29}$$

with $j \in \mathbb{Z}$, $\mathbf{k} = (k_x, k_y) \in \mathbb{Z}^2$, $\epsilon \in E = \{(0,1),(1,0),(1,1)\}$, where i denotes all possible directions $i = \{(0,1),(1,0)\}$. The function family (equation 4.29) constitutes a basis of $\left(\mathbb{L}^2\left(\mathbb{R}^2\right)\right)^2$. Hence, every vector field $\mathbf{u}$ in the function space $\left(\mathbb{L}^2\left(\mathbb{R}^2\right)\right)^2$ can be expanded into wavelet vector functions

$$\mathbf{u}(\mathbf{x}) = \sum_{i,\epsilon,j,\mathbf{k}} a_{j,\mathbf{k}}{}^{(i,\epsilon)} \boldsymbol{\psi}^{\epsilon}_{i,j,\mathbf{k}}(\mathbf{x}) \tag{4.30}$$

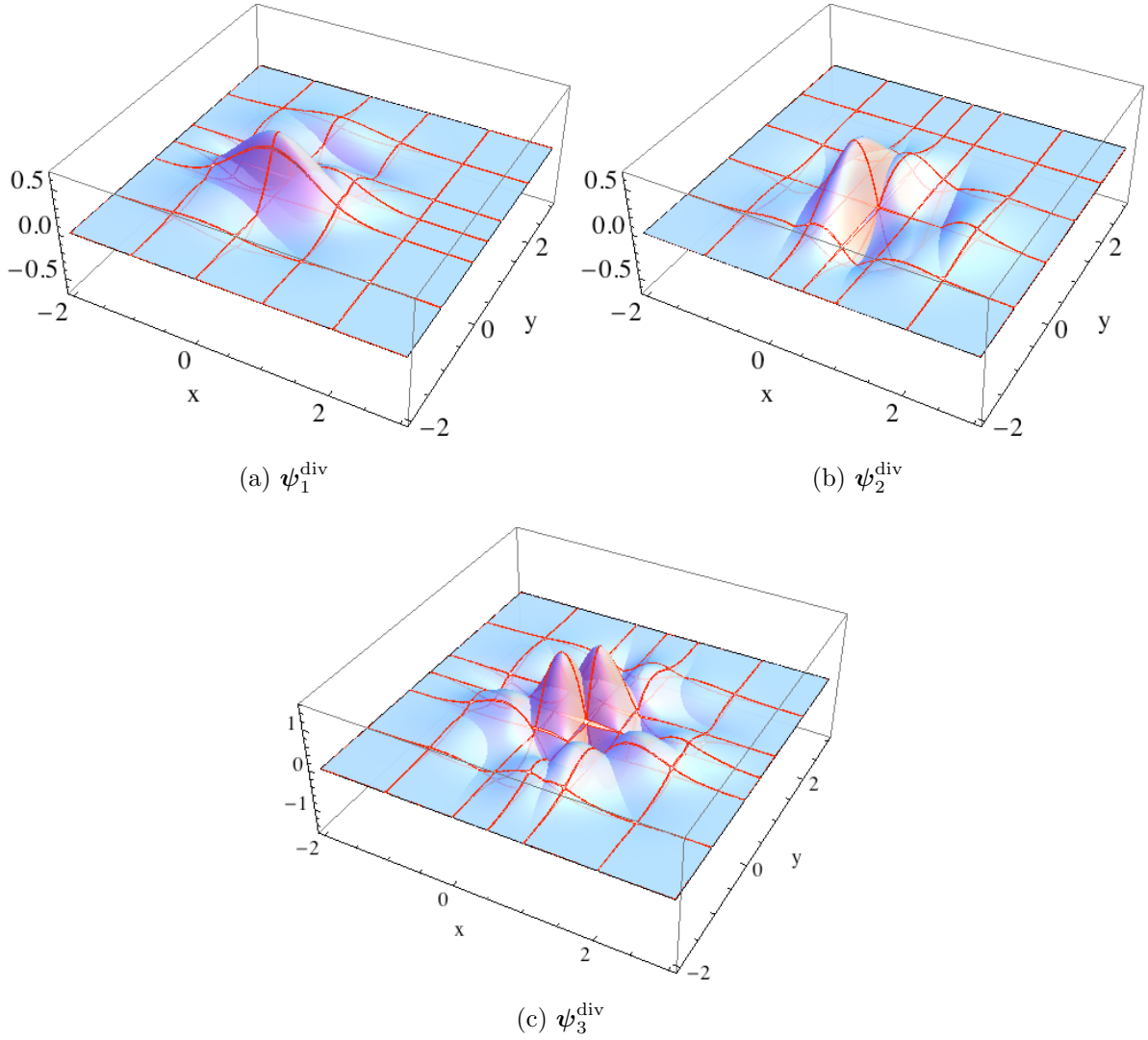
4.3.3 Two-Dimensional Divergence-Free Wavelets

The key to construct divergence-free vector wavelets lies in constructing a multivariate wavelet basis from two MRAs related by differentiation and integration, as explained above. If the MRAs are related by differentiation and integration Lemarie-Rieusset proved [12] [23] that it is possible to design a divergence-free vector wavelet basis of $\left(\mathbb{L}^2\left(\mathbb{R}^2\right)\right)^2$ from the multivariate wavelet basis of $\left(\mathbb{L}^2\left(\mathbb{R}^2\right)\right)^2$ (equation 4.29). He provided a constructive proof of the the following proposition: Let $\left(V_j^1\right)_{j\in\mathbb{Z}}$ be a one-dimensional MRA with differentiable scaling function ϕ_1 and a wavelet ψ_1, one can build a second MRA $\left(V_j^0\right)_{j\in\mathbb{Z}}$ with the scaling function ϕ_0 and wavelet ψ_0. Both MRA are related by

$$\begin{aligned} \phi_1(x)' &= \phi_0(x) - \phi_0(x-1) \\ \psi_1(x)' &= 4\psi_0(x) \end{aligned} \tag{4.31}$$

In figure 4.3 we show the scaling and wavelet functions of two such MRAs.

A basis change of the two-dimensional wavelet basis (equation 4.29) constructed from tensor products of two different MRAs verifying equation 4.31 allow to find a divergence-free basis consisting of three divergence-free vector wavelets [3] [4]. They are



(a) $\boldsymbol{\psi}_1^{\mathrm{div}}$ (b) $\boldsymbol{\psi}_2^{\mathrm{div}}$

(c) $\boldsymbol{\psi}_3^{\mathrm{div}}$

Figure 4.4: Two-dimensional divergence-free wavelets generated from two different MRAs related by differentiation and integration. In (a), (b), and (c) we depict the euclidian norm of the three divergence-free wavelets $\|\boldsymbol{\psi}_{\mathrm{div}}^{e,i}\|$. Piecewise defined spline polynomials [13] were used for the scaling and wavelet functions of the underlying MRAs, hence, the divergence-free wavelets are piecewise multivariate vector polynomials. Red lines indicated the piecewise defined polynomials.

$$\boldsymbol{\psi}_1^{\text{div}}(x,y) = \begin{bmatrix} -1/4\psi_1(x)\phi_1'(y) \\ \psi_0(x)\phi_1(y) \end{bmatrix}, \tag{4.32}$$

$$\boldsymbol{\psi}_2^{\text{div}}(x,y) = \begin{bmatrix} \phi_1(x)\psi_0(y) \\ -1/4\phi_0(x)\psi_1(y) \end{bmatrix}, \tag{4.33}$$

$$\boldsymbol{\psi}_3^{\text{div}}(x,y) = \begin{bmatrix} \psi_1(x)\psi_0(y) \\ -\psi_0(x)\psi_1(y) \end{bmatrix}. \tag{4.34}$$

In figure 4.4 we depict the three divergence-free wavelets from which one may obtain a basis by shifting and scaling. Every divergence-free field in $\mathbb{H}_{\text{div},0}(\mathbb{R}^2)$ can be decomposed into divergence-free vector wavelets

$$\mathbf{u}(\mathbf{x}) = \sum_{j\in\mathbb{Z}} \sum_{\mathbf{k}\in\mathbb{Z}^2} \sum_{\epsilon=1,2,3} c_{\epsilon,j,\mathbf{k}} \boldsymbol{\psi}_{\epsilon,j,\mathbf{k}}^{\text{div}}(\mathbf{x}) \tag{4.35}$$

The basis is derived from the wavelets by

$$\boldsymbol{\psi}_{\epsilon,j,\mathbf{k}}^{\text{div}}(\mathbf{x}) = 2^j \boldsymbol{\psi}_{\epsilon}^{\text{div}}(2^j\mathbf{x} - k) \tag{4.36}$$

4.3.4 n-Dimensional Divergence-Free Wavelets

For the n-dimensional case Lemarie-Rieusset derived the following construction principle for a divergence-free wavelet basis [12] [23] [23] [3] [4]. For the construction he used two indices e and i

$$\left(\boldsymbol{\psi}_{e,i}^{\text{div}}\right)_j(\mathbf{x}) = \begin{cases} \xi_e^{(i)}(\mathbf{x}), & j = i \\ -\frac{1}{4}\partial_{x_i}\xi_e^{(i,i_e)}(\mathbf{x}) & j = i_e \\ 0, & \text{otherwise} \end{cases} \tag{4.37}$$

where $e \in E^* = \{0,1\}^n \setminus \{(0,...,0)\}$. The integer i_e stands for the index of the first non-vanishing component of the binary vector e, thus, $i_e = \min\{l, e_l \neq 0\}$. $i \in \{1,...,n\} \setminus$

$\{i_e\}$. $j = 1, ..., n$ is the component index of $\boldsymbol{\psi}_{div}^{e,i}$. By means of a five-dimensional example, we elucidate how to build the functions $\xi_e^{(i,i_e)}$ in equation 4.37 from the scaling and wavelet functions (ϕ_0, ψ_0) and (ϕ_1, ψ_1) of two different one-dimensional MRAs related by equation 4.31. The meaning of the sub and the super index of ξ can be understood exemplarily

$$\xi_{(0,1,0,1,1)}{}^{(2,4)}(\mathbf{x}) = \phi_0(x_1)\psi_1(x_2)\phi_0(x_3)\psi_1(x_4)\psi_0(x_5) \tag{4.38}$$

As mnemonic, e is an n-dimensional binary vector consisting of the elements 0 and 1, where $0 = \phi$ and $1 = \psi$. The superscript index vector of ξ denotes the positions of factors of the product of ψ and ϕ functions (equation 4.38) which have subindex 1. In n dimensions we have $(n-1)(2^n-1)$ different basis divergence-free wavelet vectors. We label all $(n-1)(2^n-1)$ basis wavelets by $\epsilon \in \mathcal{E}_n \equiv \{1, ..., (n-1)(2^n-1)\}$, thus, we have the family

$$\left\{\boldsymbol{\psi}_\epsilon^{\text{div}}\right\}_{\epsilon\in\mathcal{E}_n}, \tag{4.39}$$

from which by integer scaling and translating in an n-dimensional integer lattice we can construct a function basis. Hence, every n-dimensional divergence-free field $\mathbf{u}$ has the following wavelet decomposition

$$\mathbf{u}(\mathbf{x}) = \sum_{j\in\mathbb{Z}} \sum_{\mathbf{k}\in\mathbb{Z}^n} \sum_{\epsilon\in\mathcal{E}} c_{\epsilon,j,\mathbf{k}} \boldsymbol{\psi}_{\epsilon,j,\mathbf{k}}^{\text{div}}(\mathbf{x}), \tag{4.40}$$

where $\boldsymbol{\psi}_{\epsilon,j,\mathbf{k}}^{\text{div}}(\mathbf{x}) = 2^{jn/2}\boldsymbol{\psi}_\epsilon^{\text{div}}(2^j\mathbf{x} - \mathbf{k})$

4.3.5 Parametrization of $\mathbb{G}$

As a result, we have a tool to construct every possible n-dimensional smooth divergence-free vector field. With this in mind, we can parametrize the group of entropy-preserving

smooth maps (equation 4.15), such that each element corresponds to a sequence of real numbers.

According to the wavelet decomposition (equation 4.35) every div-free field can be represented uniquely by its wavelet coefficients $\mathbf{c}$, therefore, instead of parametrizing an entropy-preserving transformation by a divergence-free field $\mathbf{v}$, we may use a sequence of wavelet coefficients $\mathbf{c} = (c_{\epsilon,j,\mathbf{k}})$. For convenience, we will turn to a more readable index notation by introducing the index vector

$$\mathbf{l} = (\epsilon, j, \mathbf{k}) \in \mathbb{I} \equiv \mathcal{E}_n \times \mathbb{Z} \times \mathbb{Z}^n \tag{4.41}$$

Hence, given an arbitrary flow time $t > 0$ we define a smooth entropy-preserving transformation $\mathbf{f_c}(t, \mathbf{x})$ as the solution of the following ODE at t

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{v_c}(\mathbf{x}) \\ \mathbf{v_c} &= \sum_{\mathbf{l}} c_{\mathbf{l}} \boldsymbol{\psi}_{\mathbf{l}}^{\mathrm{div}}(\mathbf{x}), \\ \mathbf{x}(0) &= \mathbf{x}. \end{aligned} \tag{4.42}$$

The group of entropy-preserving transformations (equation 4.15) becomes

$$\mathbb{G} = \left\{ \mathbf{f_c}(t, .) \mid \quad \mathbf{c} = (c_{\mathbf{l}})_{\mathbf{l} \in \mathbb{L}} \in l^2 \right\}. \tag{4.43}$$

where l^2 is the linear space of square integrable sequences.

4.4 Optimization

In section 4.1 we showed by a coarse model that the densities we are dealing with may have holes and be not very localized. Their complex topology excludes them from simple entropy estimation methods. Therefore, deforming the densities in a entropy-

preserving way, such that they assume a simpler topology is a promising approach allowing to use already established entropy estimation methods.

The wavelet representation (equation 4.42) of entropy-preserving transformations renders it possible to find wavelet coefficients $\mathbf{c}$ resulting in the "optimal deformation" $\mathbf{f_c}(t,.)$ of the density. As "optimal" we consider transformations $\mathbf{f_c}(t,.)$ deforming the density such that simple entropy estimation methods become applicable. To this end, we will introduce objective functions serving to find the optimal transformation $\mathbf{f_c}(t,.)$ by applying the steepest descent algorithm.

Lucas Parra et al. proposed a technique for finding entropy-preserving nonlinear maps producing a statistically independent representation of a probability density [14]. That means to find a nonlinear map $\mathbf{f}$ transforming a given density ρ_0 such that the resulting density $\rho = \mathbf{f}[\rho_0]$ has the following factor representation

$$\rho(\mathbf{x}) = \prod_{i=1}^{n} \rho_i(x_i), \tag{4.44}$$

where $\rho_i(x_i) = \int \rho(\mathbf{x}) \prod_{j \neq i} dx_j$ are the marginal densities of the transformed joint density ρ. In order to make $\mathbf{f}[\rho_0]$ "as factorial as possible", Parra et al. minimized its mutual information by deforming the density using implicit symplectic maps.

A factor representation of a density (equation 4.44) allows to decompose the entropy into a sum of single coordinate entropies,

$$\mathcal{S}\left[\rho(x)\right] = \sum_{i=1}^{n} \mathcal{S}\left[\rho_i(x_i)\right]. \tag{4.45}$$

Hence, the factorization reduces substantially the complexity of determining a multi-dimensional entropy problem to determining the entropy of single coordinates.

Another approach is to warp the density into a Gaussian, whose entropy can be readily estimated from its covariance matrix (reference). With this in mind, we can formulate two objectives. Either we deform the density such that it becomes "as gaussian as possible" (subsection 4.4.1) or we change its shape in a way that it becomes "as

factorial as possible " (subsection 4.4.2). On this account, we introduce two objective functions that will render it possible to find a deformation resulting in densities that achieve our before mentioned objectives sufficiently well.

4.4.1 Negentropy - Gaussianity

The fact that a Gaussian density has the largest entropy among all densities of equal covariance (reference) suggests to define a nonnegative anharmonicity measure as follows

$$\mathcal{J}[\rho] = \mathcal{S}[\rho_{\text{Gauss}}] - \mathcal{S}[\rho] \geq 0. \tag{4.46}$$

The quantity $\mathcal{J}$ is called negentropy. ρ_{Gauss} is a gaussian density with same covariance as ρ. It can be understood as a measure of the distance between an arbitrary density ρ and its Gaussian estimate. Since $\mathcal{J}[\rho]$ is zero if and only if ρ is a Gaussian, we assume that minimizing $\mathcal{J}[\mathbf{f}[\rho]]$ by deforming the density with entropy-preserving transformations $\mathbf{f}$ will yield a Gaussian-like density, which can be optimally described by a Gaussian.

The problem of deforming a given density ρ "as gaussian as possible" using entropy-preserving transformations can hence be formulated as the minimization problem

$$\mathbf{f}_{\text{min}} = \arg\left(\min_{\mathbf{f}\in\mathbb{G}} \mathcal{J}[\mathbf{f}[\rho]]\right), \tag{4.47}$$

where $\mathbb{G}$ is the set of all smooth entropy-preserving transformations (equation 4.43). Since we only allow entropy-preserving transformations, $\mathcal{S}[\mathbf{f}[\rho]]$ is constant for all $\mathbf{f} \in \mathbb{G}$, hence, we can neglect this term in the negentropy minimization. Furthermore, ρ_{Gauss} can be computed explicitly by the covariance $\mathcal{C}$ of ρ

$$\mathcal{C}_{i,j}[\rho] \equiv \langle x_i x_j \rangle_\rho - \langle x_i \rangle_\rho \langle x_j \rangle_\rho, \tag{4.48}$$

where the angular brackets $\langle . \rangle_\rho$ denote the expectation value in respect to the probability density ρ. Consequently, $\langle f(\mathbf{x}) \rangle_\rho = \int f(\mathbf{x}) \rho(\mathbf{x}) d^n\mathbf{x}$. Using the entropy expression of a Gaussian with covariance $\mathcal{C}$ (reference in text) leads to the optimization problem

$$\mathbf{f}_{\min} = \arg \left(\min_{\mathbf{f} \in \mathbb{G}} \det \mathcal{C} \left[\mathbf{f} \left[\rho \right] \right] \right). \tag{4.49}$$

4.4.2 Mutual Information - Factorizable Densities

Mutual information is a measure of statistical independence [7] [14], it is defined as follows

$$\mathcal{I} \left[\rho \right] = \sum_{i=1}^{n} \mathcal{S} \left[\rho_i \right] - \mathcal{S}[\rho] \geq 0, \tag{4.50}$$

where $\rho_i(x_i)$ are the marginal densities of the joint density $\rho(\mathbf{x})$ and $\mathcal{S}[\rho]$ is the entropy of ρ. Mutual information vanishes if and only if all components of the random variable $\mathbf{x}$ are statistically independent according to density $\rho(\mathbf{x})$. Equivalently, statistical independence occurs when the joint density factorizes in a product of its marginal densities

$$\rho(\mathbf{x}) = \prod_{i=1}^{n} \rho_i(x_i) \Leftrightarrow \mathcal{I}[\rho] = 0. \tag{4.51}$$

In order to transform a given density ρ to a density that is "as factorial as possible", we have to search a transformation minimizing mutual information

$$\mathbf{f}_{\min} = \arg \left(\min_{\mathbf{f} \in \mathbb{G}} \mathcal{I} \left[\mathbf{f} \left[\rho \right] \right] \right). \tag{4.52}$$

Lucas Parra et al. suggested to minimize an upper bound to avoid the computationally intensive task of measuring single coordinate-entropies [14]. An upper bound that is easy to minimize is

$$\mathcal{I}[\rho] \leq -\mathcal{S}[\rho] + \frac{n}{2}\ln(2\pi e) + \frac{1}{2}\sum_{i=1}^{n}\sigma_i^2 \tag{4.53}$$

Instead of minimizing the individual coordinate entropies $\mathcal{S}[\rho(x_i)]$ the problem simplifies to the minimization of the variance σ_i^2. Using only second moments might seem as a strong simplifications. It leads, however, to a computationally efficient solution.

Using the identity $\operatorname{tr}\mathcal{C} = \sum_{i=1}^{n}\sigma_i^2$ and neglecting the constant terms in equation 4.53 simplifies the minimizing of the upper bound of mutual information to

$$\mathbf{f}_{\min} = \arg\left(\min_{\mathbf{f}\in\mathbb{G}} \operatorname{tr}\,\mathcal{C}\left[\mathbf{f}\left[\rho\right]\right]\right). \tag{4.54}$$

It can be proven that a circular shaped Gaussian extremizes the above minimization problem [14], hence, if the transformations are flexible enough, the minimization tends to produce circular Gaussian shaped functions.

4.4.3 Approximation of Objective Functions

An MD simulation yields a sampled trajectory that can be considered as a series of snapshots of possible solvent configurations each drawn from the configurational density ρ (reference). We denote the trajectory by

$$\mathbf{X} = \left\{\mathbf{x}^{(m)} \mid \quad m = 1, ..., M\right\}. \tag{4.55}$$

We transform each configuration by an entropy-preserving transformation $\mathbf{f}$ and denote it by

$$\mathbf{Y} = \mathbf{f}\left(\mathbf{X}\right) \equiv \left\{\mathbf{f}\left(\mathbf{x}^{(m)}\right) \mid \quad m = 1, ..., M\right\}. \tag{4.56}$$

Therefore, each transformed snapshot $\mathbf{y}_m = \mathbf{f}(\mathbf{x}_m)$ is drawn from the transformed density $\mathbf{f}[\rho](\mathbf{y}) = \rho\left(\mathbf{f}^{-1}(\mathbf{y})\right)$ (reference).

With this we may approximate the covariance $\mathcal{C}[\rho]$ (reference) from sample points of the trajectory by

$$\mathbf{Cov}_{ij}(\mathbf{X}) \equiv \langle x_i x_j \rangle - \langle x_i \rangle \langle x_j \rangle \tag{4.57}$$

where $\langle f(\mathbf{x}) \rangle = \frac{1}{M} \sum_{i=1}^{M} f\left(\mathbf{x}^{(m)}\right)$.

Given the trajectory $\mathbf{X}$, we estimate the objective functions (equation 4.54 and equation 4.49) using the approximated covariance $\mathbf{Cov}$

$$\min_{\mathbf{c} \in l^2} \quad \mathrm{tr}\left[\mathbf{Cov}\left(\mathbf{f_c}\left(t, \mathbf{X}\right)\right)\right] \tag{4.58}$$

or

$$\min_{\mathbf{c} \in l^2} \quad \det\left[\mathbf{Cov}\left(\mathbf{f_c}\left(t, \mathbf{X}\right)\right)\right]. \tag{4.59}$$

Additionally, we exploit the fact that each entropy preserving map corresponds uniquely to a l^2-sequence $\mathbf{c}$ (equation 4.43).

For a given trajectory $\mathbf{X}$ we may find an entropy-preserving transformation by searching the corresponding sequence of wavelet coefficients that transforms $\mathbf{X}$ to a trajectory $\mathbf{Y}$ whose underlying density is more Gaussian.

4.4.4 Wavelet Coefficients - Compression Effects

Obtaining a more gaussian shaped trajectory requires to find an infinite number of square integrable wavelet coefficients $\mathbf{c} = \{c_\mathbf{l}\}_{\mathbf{l} \in \mathbb{I}}$ corresponding to an entropy-preserving transformation $\mathbf{f_c}(t, .)$ minimizing equation 4.58 or 4.59. However, it is infeasible to numerically perform a minimization process in an infinite parameter space of wavelet coefficients. Therefore, we will present a heuristic method for choosing a finite number of wavelet coefficients corresponding to an entropy-preserving $\mathbf{f_c}(t, .)$. We make a choice on the wavelet coefficients such that we take the length scale and the size of the given trajectory into account.

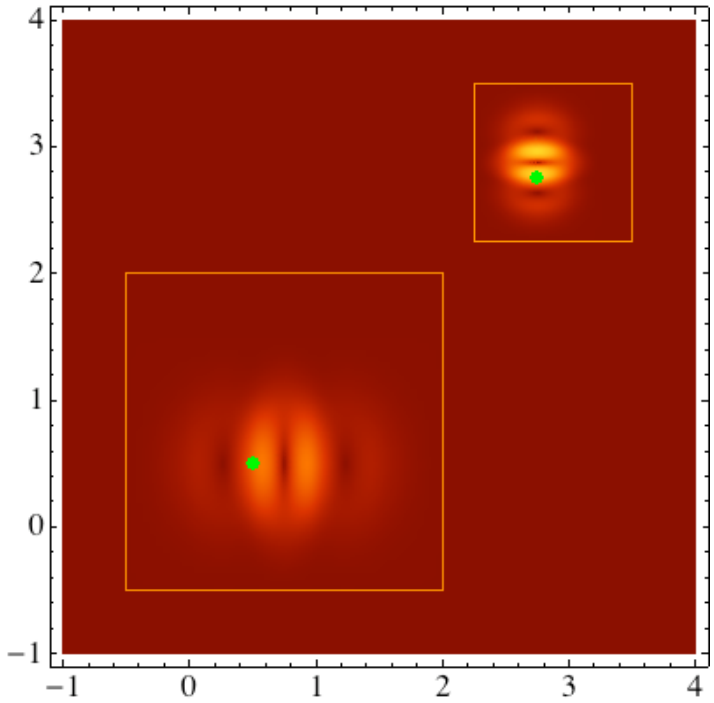


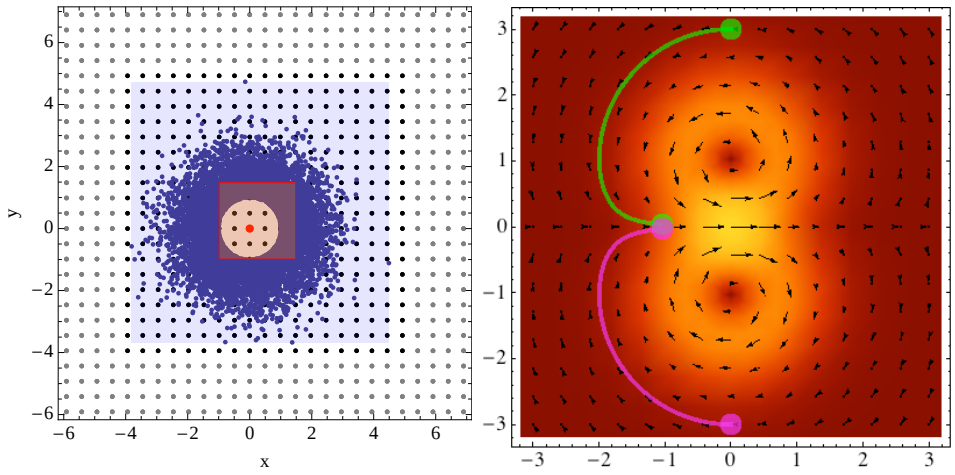
Figure 4.5: Density plot of the norm of two divergence-free spline wavelet $\boldsymbol{\psi}^{\text{div}}_{1,1,(1,1)}$ and $\boldsymbol{\psi}^{\text{div}}_{1,2,(11,11)}$ with compact support (reference). The basic wavelet $\boldsymbol{\psi}_1$ is translated to the grid points $2^{-1}(1,1)$ and $2^{-2}(11,11)$ (green dots). The length scale of $\boldsymbol{\psi}^{\text{div}}_{1,1,(1,1)}$ is smaller than the length scale of $\boldsymbol{\psi}^{\text{div}}_{1,2,(11,11)}$ indicated by their support (orange square).

To start with, we elucidate the meaning of the wavelet indices. Every vector index $\mathbf{l}$ (equation 4.41) of a wavelet coefficient $c_{\mathbf{l}} = c_{\epsilon,\mathbf{j},\mathbf{k}}$ corresponds to an n-dimensional divergence-free wavelet $\boldsymbol{\psi}^{\text{div}}_{\epsilon,j,\mathbf{k}}$. This wavelet is constructed by scaling and translating a basic wavelet

$$\boldsymbol{\psi}^{\text{div}}_{\epsilon,j,\mathbf{k}}(\mathbf{x}) = 2^{jn/2}\boldsymbol{\psi}^{\text{div}}_{\epsilon}\left(2^{j}\mathbf{x} - \mathbf{k}\right), \tag{4.60}$$

hence, the support of $\boldsymbol{\psi}^{\text{div}}_{\epsilon,j,\mathbf{k}}$ — the domain where this wavelet is non-zero — is

$$\operatorname{supp}\boldsymbol{\psi}^{\text{div}}_{\epsilon,j,\mathbf{k}} = 2^{-j}\operatorname{supp}\boldsymbol{\psi}^{\text{div}}_{\epsilon} + 2^{-j}\mathbf{k}. \tag{4.61}$$



(a) Indices (b) Compression Effects

Figure 4.6: (a) Illustrated are the grid of $\mathbf{k}$ values for a fixed j for a given two-dimensional trajectory (blue dots) The resolution level of wavelets j is chosen such that $2^{-j} = 20 r_{\mathrm{NN}}$, where r_{NN} is the mean nearest neighbor distance of the trajectory. The blue transparent box is the bounding box of trajectory, the red square is the support of a single wavelet, and the black and gray points indicate the grid $2^{-j}\mathbf{k}$. (b) When the length scale of wavelets is much smaller than nearest neighbor distance of the trajectory, the resulting entropy-preserving transformation moves single points closer together.

The above formula reveals that the basic wavelet $\boldsymbol{\psi}_{\epsilon}^{\mathrm{div}}$ is located at the grid point $2^{-j}\mathbf{k}$ and its spatial resolution is determined by j. The larger j the smaller the length scale of the basic wavelet $\boldsymbol{\psi}_{\epsilon}^{\mathrm{div}}$ and vice versa. In figure 4.5 we have illustrated the meaning of j and $\mathbf{k}$ for two divergence-free spline wavelet with compact support.

The choice of the index j depends on the considered trajectory $\mathbf{X}$. For a given trajectory $\mathbf{X}$ we suggest to determine the nearest neighbor distance r_{NN} of $\mathbf{X}$ to obtain a measure of the length scale of the trajectory and to choose the resolution level j of the wavelets larger than the length scale of $\mathbf{X}$. If the length scale of the wavelets is much

smaller than the length scale of the trajectory we may have compression effects, as we illustrate in figure 4.6. The reason is that the wavelet "sees" the trajectory as points and not as a continuous density. Hence, a minimization with wavelets of a length scale smaller than the length scale of the trajectory would compress the trajectory, since minimizing the variances (equation 4.58) favors a more dense trajectory. To avoid such artifact, we suggest to choose a fixed resolution level j such that it satisfies

$$2^{-j} > r_{\mathrm{NN}} \tag{4.62}$$

In a more general consideration, we may can choose $j_{\min} \leq j \leq j_{\max}$. $j_{\max}$ accounts for the length scale of $\mathbf{X}$ and $j_{\min}$ for the size of $\mathbf{X}$. It does not make sense to choose resolution levels that exceed the geometric scale of the trajectory, as they would give rise to simple rotations and translations, which don not lead to deformation. However, different j would increase the computational costs, since more wavelet coefficient has to be considered for minimization.

Once the resolution level j is determined, we choose all grid points $\mathbf{k}$ such that they are within the bounding box of the given trajectory $\mathbf{X}$ as depicted in figure 4.6. In case of two dimensions and if $[x_{1,\min}, x_{1,\max}] \times [x_{2,\min}, x_{2,\max}]$ is the bounding box of $\mathbf{X}$, we choose

$$\begin{aligned} k_{1,\min} &= \left\lfloor 2^j x_{1,\min} \right\rfloor, \\ k_{2,\min} &= \left\lfloor 2^j x_{2,\min} \right\rfloor, \\ k_{1,\max} &= \left\lceil 2^j x_{1,\max} \right\rceil, \\ k_{2,\max} &= \left\lceil 2^j x_{2,\max} \right\rceil, \end{aligned} \tag{4.63}$$

where $\lceil x \rceil$ indicate the smallest integer n with $n \geq x$, and $\lfloor x \rfloor$ the largest integer n such that $n \leq x$. We select all grid points $\mathbf{k}$ with $k_1 = k_{1,\min}, ..., k_{1,\max}$ and $k_2 = k_{2,\min}, ..., k_{2,\max}$.

We always choose all possible basic wavelets, hence we don't make a specific choice on the index ϵ since each of the $(n-1)(2^n-1)$ divergence-free basis wavelets $\boldsymbol{\psi}_\epsilon^{\text{div}}$ acts in different directions of the considered configurational space.

As we have seen, we can choose indices $\mathbf{l} = (\epsilon, j, \mathbf{k})$ and, hence, wavelet coefficients $c_\mathbf{l}$ such that the corresponding divergence-free field $\mathbf{v_c}$ (equation 4.42) consists of a finite number of wavelets $\boldsymbol{\psi}_\mathbf{l}^{\text{div}}$ taking the length scale and the size of the given trajectory $\mathbf{X}$ into consideration. We will denote these indices by $\boldsymbol{\Lambda}$. For instance, for the two dimensional case we have

$$\boldsymbol{\Lambda} = \{(\epsilon, j, (k_1, k_2)) \mid \quad \epsilon = 1, ..., 3 \; k_1 = k_{1,\min}, ..., k_{1,\max} \; k_2 = k_{2,\min}, ..., k_{2,\max}\}, \quad (4.64)$$

where j is fixed and satisfies condition 4.62 and $k_{1,\min}$, $k_{1,\max}$, $k_{2,\min}$ and $k_{2,\max}$ are determined according to equation 4.63.

4.4.5 Steepest Descent

By means of steepest descent we will exemplarily show how to minimize the objective function equation 4.58 with a gradient based optimization scheme. For convenience Q will refer to the objective function

$$Q(\mathbf{c}) \equiv \text{tr}\left[\mathbf{Cov}\left(\mathbf{f_c}\left(\mathbf{X}\right)\right)\right] \quad (4.65)$$

The idea of steepest descent is to construct a convergent sequence of wavelet coefficient sets $\left\{\mathbf{c}^{(1)}, \mathbf{c}^{(2)}, ...\right\}$ satisfying $Q\left(\mathbf{c}^{(n+1)}\right) \leq Q\left(\mathbf{c}^{(n)}\right)$. Each $\mathbf{c}^{(n)}$ is iteratively defined by

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \gamma_n \nabla_\mathbf{c} Q\left(\mathbf{c}^{(n)}\right). \quad (4.66)$$

γ_n is chosen such that $g_n(\gamma) = Q\left(\mathbf{c}^{(n)} - \gamma \nabla_\mathbf{c} Q\left(\mathbf{c}^{(n)}\right)\right)$ assumes its minimum at γ_n. Finding a minimum of $g_n(\gamma)$ is left to a simple line search algorithm [16]. The sequence

of wavelet coefficient sets will converge to the wavelet coefficients at which Q has a (local) minimum.

A crucial part of gradient based minimum search algorithms is to have the gradient of the objective function preferably as analytic expression. Using the short notation $\partial_{\mathbf{l}} = \frac{\partial}{\partial c_{\mathbf{l}}}$ and the transformed trajectory $\mathbf{Y} = \mathbf{f_c}(t, \mathbf{X})$ the gradient of Q is

$$\partial_{\mathbf{l}} Q(\mathbf{c}) = \text{tr}\left[\partial_l \mathbf{Cov}\left(\mathbf{Y}\right)\right] \tag{4.67}$$

By simple algebra we ca easily derive the gradient of $\text{Cov}(\mathbf{Y})$.
Let $\partial_{\mathbf{l}}\mathbf{Y} = \left\{\partial_{\mathbf{l}}\mathbf{y}^{(m)}, \quad m = 1, ..., M\right\}$ be the set of gradients of the M transformed configurational vectors of the trajectory $\mathbf{X}$. The Gradient of the covariance matrix in respect to the wavelet coefficient $\mathbf{c_l}$ is

$$\partial_l \mathbf{Cov}\left(\mathbf{Y}\right) \equiv \mathbf{A}\left(\mathbf{Y}, \partial_l \mathbf{Y}\right) + \mathbf{A}\left(\mathbf{Y}, \partial_l \mathbf{Y}\right)^{\mathrm{T}}, \tag{4.68}$$

where the matrix $\mathbf{A}$ is defined as follows

$$\mathbf{A}\left(\mathbf{Y}, \partial_l \mathbf{Y}\right) \equiv \left\langle \partial_l \mathbf{Y} \otimes \mathbf{Y} \right\rangle - \left\langle \partial_l \mathbf{Y} \right\rangle \otimes \left\langle \mathbf{Y} \right\rangle. \tag{4.69}$$

The angular brackets denote the average. On the right side of equation 4.69, the tensor product $\otimes$ of the first term is to be understood as follows

$$\left\{\mathbf{a}_1, ..., \mathbf{a}_M\right\} \otimes \left\{\mathbf{b}_1, ..., \mathbf{b}_M\right\} \equiv \left\{\mathbf{a}_1 \otimes \mathbf{b}_1, ..., \mathbf{b}_M \otimes \mathbf{b}_M\right\}. \tag{4.70}$$

The time evolution of the gradient $\partial_{\mathbf{l}}\mathbf{y}$ (equation 4.41) is described by the following ordinary differential equation (appendix 8.3)

$$\partial_t \partial_{\mathbf{l}} \mathbf{y} = \boldsymbol{\psi}_{\mathbf{l}}^{\text{div}}\left(\mathbf{y}\right) + \mathbf{J}_{\mathbf{v_c}}\left(\mathbf{y}\right) \cdot \partial_{\mathbf{l}} \mathbf{y} \tag{4.71}$$

where $\mathbf{J}_{\mathbf{v}_c}$ is the Jacobian matrix of the divergence-free field $\mathbf{v_c}$. And $\boldsymbol{\psi}_{\mathbf{l}}^{\mathrm{div}}$ is the divergence-free wavelet that corresponds with the wavelet coefficient $\mathbf{c_l}$ in the wavelet expansion of $\mathbf{v_c}$ (equation 4.42). At time $t = 0$ the gradient is initialized with

$$\partial_{\mathbf{l}}\mathbf{y} \mid_{t=0} = 0. \tag{4.72}$$

The analytical expression of the gradient enables us to employ a gradient based minimization scheme we apply to the objective function to determine the wavelet coefficients of an entropy-preserving transformation that yield a trajectory whose entropy can be simply estimated by its covariance matrix.

4.5 Algorithm

Given a trajectory $\mathbf{X}$, where each snapshot is n dimensional, we need n dimensional entropy-preserving maps to deform the whole trajectory. However, the mere number of $(n-1)(2^n-1)$ basic wavelets $\{\boldsymbol{\psi}_\epsilon\}_{\epsilon\in\{1,\dots,(n-1)(2^n-1)\}}$ that serve to build n-dimensional entropy-preserving maps, renders it infeasible to perform an optimization process in the resulting huge parameter space of wavelet coefficients (reference to how to chose wavelet coefficients). Therefore, we will develop an iterative optimization algorithm, that only performs optimization in two dimensions, where we have only 3 basis wavelets. We follow and idea developed by Oliver Lange in the framework of Full Correlation Analysis (reference). His algorithm aims to decrease the correlation between atomic displacements by minimizing mutual information using rotations. He suggested to split the multidimensional minimization problem into many two-dimensional minimization problems. For this purpose we introduce two dimensional entropy-preserving maps that transform the configuration vector $\mathbf{x} = (x_1, ..., x_i, ..., x_j, ..., x_N)^T$ as follows

$$\mathbf{f}_{\mathbf{c}}^{(i,j)}(t, \mathbf{x}) \equiv (x_1, ..., y_i, ..., y_j, ..., x_N)^T \tag{4.73}$$

with the two-dimensional transformation $\mathbf{f_c}(t,.)$ affecting only the i^{th} and the j^{th} component of $\mathbf{x}$ denoted by plane (i,j)

$$(y_i, y_j) = \mathbf{f_c}\left(t, (x_i, x_j)^T\right). \tag{4.74}$$

Having in mind that the set of entropy-preserving transformations $\mathbb{G}$ (equation 4.43) is a group, we try to find an entropy-preserving transformation that minimizes the upper bound of mutual information (equation 4.58) by composing the transformation of a multitude of two-dimensional entropy-preserving maps

$$\mathbf{f}(t,.) = \prod_{k=1}^{K_{\max}} \mathbf{f}_{\mathbf{c}_k}^{(i_k,j_k)}(t,.). \tag{4.75}$$

The idea is to render the initial trajectory $\mathbf{X}$ more Gaussian at every iteration step k using two-dimensional entropy preserving transformations, such that the entropy of $\mathbf{X}$ eventually can be approached by a gaussian. We suggest to obtain each $\mathbf{f}_{\mathbf{c}_k}^{(i_k,j_k)}(t,.)$ by the following iterative scheme: We start by choosing a "promising" plane (i_1, j_1) of the trajectory $\mathbf{X}$, and determine $\mathbf{c}_1$ by minimizing the upper bound of mutual information in this plane. Finally, we apply the thus obtained transformation $\mathbf{f}_{\mathbf{c}_1}^{(i_1,j_1)}(t,.)$ to the trajectory yielding the transformed trajectory $\mathbf{X}_1$. Then we repeat the process and obtain a sequence $\{\mathbf{X}_2, \mathbf{X}_3, ...\}$ until proceeding does not yield further minimization. The iteration scheme is

$$\mathbf{X_k} = \mathbf{f}_{\mathbf{c}_k}^{(i_k,j_k)}\left(t, \mathbf{X}_{k-1}\right), \tag{4.76}$$

where $k = 1, ..., K_{\max}$ and $\mathbf{X}_0 = \mathbf{X}$. $\mathbf{f}_{\mathbf{c}_k}^{(i_k,j_k)}(t,.)$ is chosen such that it minimizes the upper bound of mutual information, hence, we obtain a sequence of deformed trajectories $\mathbf{X}_k$ satisfying

$$0 \leq \operatorname{tr}\left\{\mathbf{Cov}(\mathbf{X}_{k+1})\right\} \leq \operatorname{tr}\left\{\mathbf{Cov}(\mathbf{X}_k)\right\} \tag{4.77}$$

what implies that

$$0 \leq \mathrm{tr}\left\{\mathbf{Cov}(\mathbf{X}_k)\right\} \leq \mathrm{tr}\left\{\mathbf{Cov}(\mathbf{X})\right\} \tag{4.78}$$

We choose the planes (i_k, j_k) heuristically such that we keep the number of two-dimensional minimizations small in equation 4.75. $K_{\max}$ denotes the number of iterations that are carried out until not further minimization is possible. Since we except that minimization will yield a high loss of mutual information, we start with planes (i, j) featuring high pairwise mutual information

$$I_{ij} \equiv \mathcal{I}\left[\rho_{(i,j)}\right], \tag{4.79}$$

where $\rho_{(i,j)}$ is the marginal density

$$\rho_{(i,j)}(x_i, x_j) = \int \rho(\mathbf{x}) \prod_{l \neq i,j} dx_l \tag{4.80}$$

The heuristic selection of planes requires to numerically determine pairwise mutual information from a transformed trajectory. We can efficiently estimate the occurring one and two-dimensional densities, which are required for the entropy estimate, simply using an histogram estimator [8] [10].

Additionally, redundant minimization evaluations of already visited planes is avoided by using a marker matrix $\mathbf{m}$. Each entry m_{ij} indicates the degree of necessity to minimize plane (i, j). We initialize each entry of $\mathbf{m}$ with 1. We set m_{ij} to zero after minimization. Since applying a deformation $\mathbf{f}_{\mathbf{c}}^{(i,j)}$ to plane (i, j) increases the probability that an already marked plane (i, k) or (k, j), $k \neq i, j$ needs further minimization, all respective markers are increased by the norm of the wavelet coefficients $\|\mathbf{c}\| \equiv \sqrt{\sum_{\mathbf{l}} c_{\mathbf{l}}^2}$, to put this planes in the waiting queue for further minimization. Finally, at every iteration step k we chose $(i_k, j_k) = \underset{i,j}{\mathrm{argmax}}\left(m_{ij} I_{ij}\right)$

For a clear comprehension we summarize the algorithm in a pseudo code. The input parameters are a trajectory $\mathbf{X} = \left\{\mathbf{x}^{(m)}, \quad m = 1, ..., M\right\}$ of M snapshots. Each snapshot $\mathbf{x}^{(m)}$ is D-dimensional. With $\mathbf{X}_i = \left\{x_i^{(m)}, \quad m = 1, ..., M\right\}$ we denote the projection of $\mathbf{X}$ on i^{th} unit vector. The pairwise mutual information can be approximated with an histogram estimator [10] from the sample points denoted by $\mathcal{I}\left[\mathbf{X}_i, \mathbf{X}_j\right]$

Initialize pairwise mutual information matrix $\mathbf{I}$
and marker matrix $\mathbf{m}$
for $i = 1$ to D **do**
  for $= 1$ to D **do**
    $m_{ij} = 1$
    $I_{ij} = \mathcal{I}\left[\mathbf{X}_i, \mathbf{X}_j\right]$
  end for
end for

while $\sum_{i,j} m_{ij} I_{ij} > \epsilon$ **do**
  Choose "promising" plane
  $(i, j) = \underset{i \neq j}{\operatorname{argmax}} \left(m_{ij} I_{ij}\right)$

  Find wavelet coefficients of entropy-preserving transformation minimizing upper bound of mutual information of $\mathbf{X}$
  $\mathbf{c}_{\min} = \min_{\mathbf{c}} \left\{\operatorname{tr}\left(\operatorname{Cov}\left[\mathbf{f}_{\mathbf{c}}^{(i,j)}(\mathbf{X})\right]\right)\right\}$

  Deform plane such that it gets more gaussian
  $\mathbf{X} = \mathbf{f}_{\mathbf{c}_{\min}}^{(i,j)}(\mathbf{X})$

  Update marker matrix $\mathbf{m}$
  for $k = 1$ to D **do**

$m_{ik} = m_{ik} + \|c_{\min}\|$

$m_{kj} = m_{kj} + \|c_{\min}\|$

end for

$m_{i,j} = 0$

Update pairwise mutual information matrix $\mathbf{I}$

for $k = 1$ to D **do**

if $k \neq i$ **then**

$I_{ik} = \mathcal{I}[\mathbf{X}_i, \mathbf{X}_k]$

end if

if $k \neq j$ **then**

$I_{kj} = \mathcal{I}[\mathbf{X}_k, \mathbf{X}_j]$

end if

end for

end while

In following we will refer to the algorithm of searching an entropy-preserving transformation to as **EPTM** (**E**ntropy **P**reserving **T**ransformation **M**ethod).

5 Applications

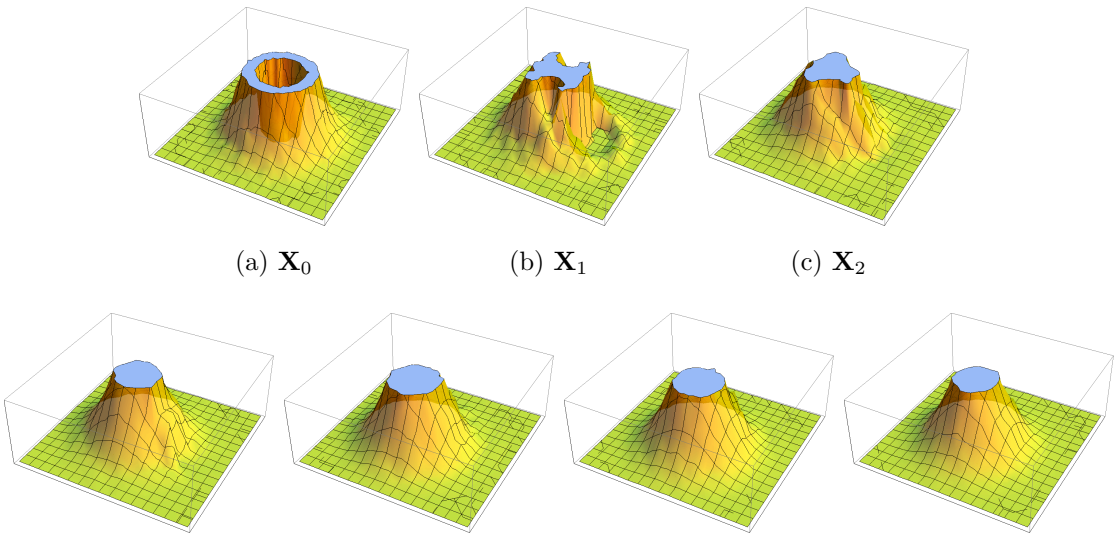
In section 4.1 we characterized the topology of solvent-protein densities and showed that the repulsive interaction between solvent molecules and the interaction between the solvent and the protein gives to parts of the configurational space with vanishing density to which we refer as holes. Applying Reinhard's permutation reduction (section 3.2) to densities with holes yields a more compact density that still has holes at the surface and in the interior. Hence, a simple Gaussian approximation of the density is not possible. To assess our newly developed **EPTM** we will apply it to a few artificial densities with features typical for solvent densities (section 4.1) and whose entropies are analytically known. We will show that we can deform these densities such that they become more Gaussian shaped while at the same time the entropy is conserved.

To this end we have developed a command line tool — *g_entropyestimate* — that read in a trajectory and searches via conjugated gradient descent [16] for an entropy-preserving transformation that renders the trajectory more Gaussian and finally applied the found transformation to the trajectory and estimated the entropy using a Gaussian. Since constructing entropy-preserving maps requires to solve a differential equation (equation 4.42) we used a simple implicit midpoint scheme [16] to approximate the solution of the differential equation.

5.1 Two-Dimensional Densities

In the next two sections we will prove that entropy-preserving maps can remove holes in the density away and deforming the density into a circular Gaussian. For this purpose, we will apply EPTM to two different densities, one characterized by a hole within the density and another by a hole at the surface.

5.1.1 Hole in the Center



(a) $\mathbf{X}_0$ (b) $\mathbf{X}_1$ (c) $\mathbf{X}_2$

(d) $\mathbf{X}_3$ (e) $\mathbf{X}_4$ (f) $\mathbf{X}_5$ (g) $\mathbf{X}_6$

Figure 5.1: Density estimates of transformed trajectories $\mathbf{X}_i$. The initial trajectory $\mathbf{X}_0$ is iteratively transformed by EPTM. The trajectories converge to a circular Gaussian shaped trajectory. For better visualization we additionally display the cross sections of the densities (light blue) by using an according plot range.

We commence with a two-dimensional density featuring a hole in the center:

$$\rho(x,y) = \frac{\sqrt{e}}{2\pi} \begin{cases} \exp\left(-\frac{x^2+y^2}{2}\right), & x^2+y^2 > 1 \\ 0, & \text{otherwise} \end{cases}. \tag{5.1}$$

The entropy $\mathcal{S}$ is $1 + \ln(2\pi) \approx 2.838$. From this density we chose $50,000$ random vectors constituting the trajectory $\mathbf{X}_0$. We iteratively applied EPTM to the initial trajectory and obtained a sequence of trajectories $\{\mathbf{X}_0, ..., \mathbf{X}_6\}$ that converges towards a Gaussian (figure 5.1).

To avoid compression artifacts (subsection 4.4.4) we chose wavelets that allow to construct entropy-preserving maps with a length scale much larger than the mean nearest neighbor distance r_{NN} of the trajectory (inequality 4.62). Instead of choosing an integer value for the wavelet index j (equation 4.41) we allow it to be a real number since we want to have continuous values for the length scale $s = 2^{-j}$.

	r_{NN}	$s = 2^{-j}$	S_{Gauss}	$S_{\text{Gauss}} - S$
$\mathbf{X}_0$	1.132×10^{-2}	80×10^{-2}	3.249	4.11×10^{-1}
$\mathbf{X}_1$	1.133×10^{-2}	60×10^{-2}	3.020	1.82×10^{-1}
$\mathbf{X}_2$	1.129×10^{-2}	80×10^{-2}	2.940	1.02×10^{-1}
$\mathbf{X}_3$	1.127×10^{-2}	80×10^{-2}	2.921	0.84×10^{-1}
$\mathbf{X}_4$	1.141×10^{-2}	80×10^{-2}	2.916	0.78×10^{-1}
$\mathbf{X}_5$	1.163×10^{-2}	80×10^{-2}	2.917	0.79×10^{-1}
$\mathbf{X}_6$	1.158×10^{-2}	80×10^{-2}	2.916	0.78×10^{-1}

Table 5.1: Entropy and mean nearest neighbor distances of transformed trajectories. S is the real entropy and S_{Gauss} the entropy estimate using a Gaussian. s is the length scale of the used entropy-preserving transformations.

Table 5.1.1 the mean nearest neighbor distances r_{NN} revealed a negligible dilatation

between the initial trajectory $\mathbf{X}_0$ and the converged trajectory $\mathbf{X}_6$ of $\Delta r_{\mathrm{NN}} = 2.6 \times 10^{-4}$.

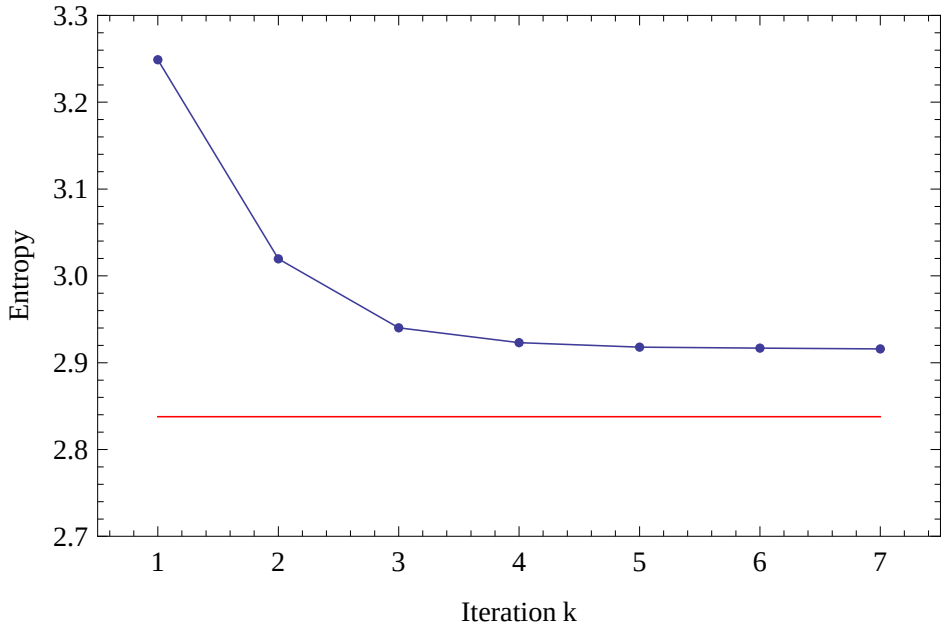


Figure 5.2: Entropy estimate with covariance matrix at each iteration step k (blue line). The real entropy is $S = 2.838$ (red line). At each iteration step we have minimized the upper bound of mutual information

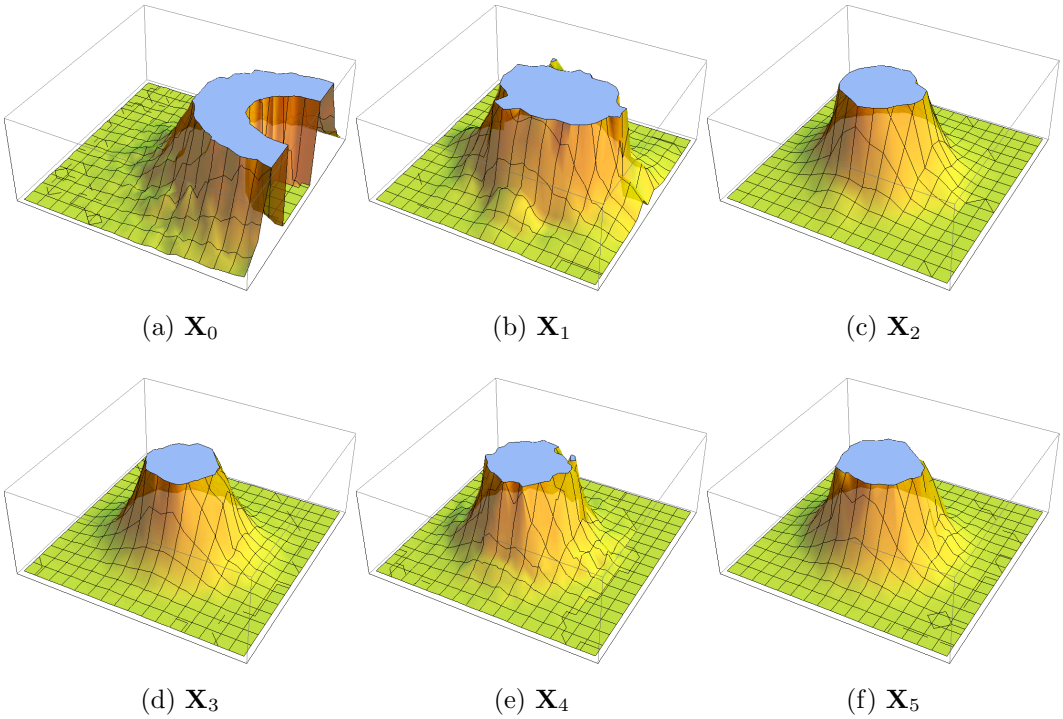
We estimated the entropy using a Gaussian density approximation of the trajectories (equation 4.57) at each iteration step k:

$$\mathcal{S}_{\mathrm{Gauss}}(\mathbf{X}) = 1 + \frac{1}{2} \ln \left(4\pi^2 \det \mathbf{Cov}(\mathbf{X}) \right) . \tag{5.2}$$

Applying EPTM iteratively to $\mathbf{X}_0$ renders the given density more gaussian and, thus, significantly improves the entropy estimate as it can be observed in figure 5.2. The deviation of the estimated entropy from the analytic value drops down (table 5.1.1). However, the method fails to reproduce the exact analytic value of the entropy $\mathcal{S} = 1 + ln(2\pi) \approx 2.838$. Estimating the entropy with an histogram estimator with optimal

bin size as described in [10] gives $\mathcal{S}_{\text{hist}} = 2.926$. $EPTM$ with an Gaussian entropy estimate converges to $\mathcal{S}_{\text{Gauss}} = 2.916$ (figure 5.2).

5.1.2 Hole at the Surface



(a) $\mathbf{X}_0$ (b) $\mathbf{X}_1$ (c) $\mathbf{X}_2$

(d) $\mathbf{X}_3$ (e) $\mathbf{X}_4$ (f) $\mathbf{X}_5$

Figure 5.3: Density estimates of transformed trajectories $\mathbf{X}_i$ each consisting of $50,000$ two-dimensional sample points. The initial trajectory $\mathbf{X}_0$ is iteratively transformed by entropy-preserving maps such that the upper bound of mutual information is minimized (equation 4.58). The trajectories converges to a circular Gaussian shaped trajectory.

A simple two-dimensional function exemplifying a density with a hole at the surface is

$$\rho(x,y) = \frac{\sqrt{e}}{\pi} \begin{cases} \exp\left(-\frac{x^2+y^2}{2}\right), & x^2+y^2 > 1 \wedge x < y \\ 0, & \text{otherwise} \end{cases}. \tag{5.3}$$

The analytically calculated entropy is $\mathcal{S}[\rho] = 1 + \ln(\pi) \approx 2.145$. We iteratively applied EPTM to the initial trajectory $\mathbf{X}_0$ consisting of $50,000$ random vectors drawn from density 5.3 and obtained a sequence of trajectories $\{\mathbf{X}_0, ..., \mathbf{X}_5\}$. EPTM improves the

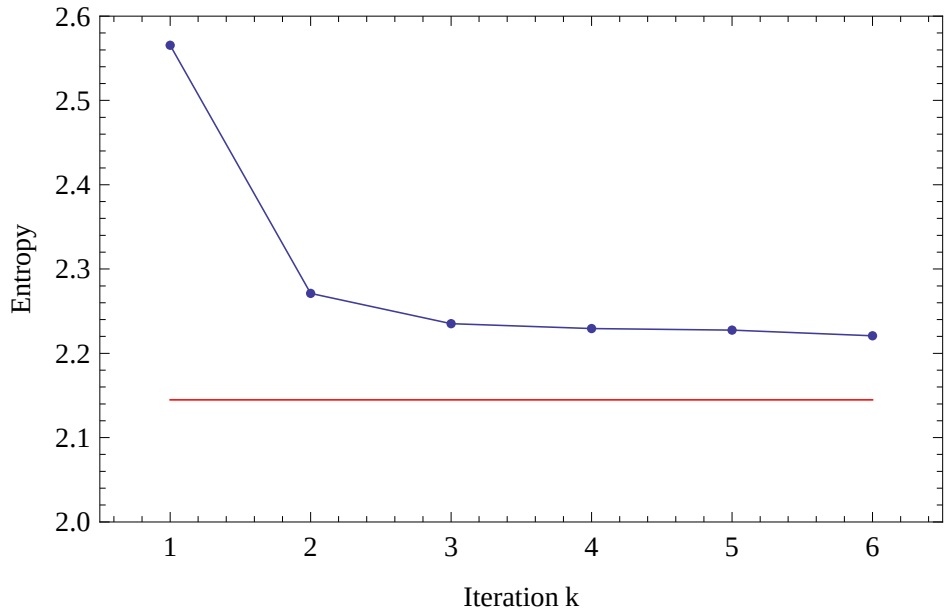


Figure 5.4: Gaussian entropy estimate of iteratively transformed trajectory.

entropy estimate since it deforms the initial trajectory to a more Gaussian shaped one (figure 5.3 and 5.4). The difference of the mean nearest neighbor distance between the initial trajectory $\mathbf{X}_0$ and the transformed trajectory $\mathbf{X}_5$ is negligible (table 5.2). EPTM fails to transform the initial trajectory such that we can extract the exact analytic value with a Gaussian entropy estimate. Estimating the entropy using an histogram estimator yields $\mathcal{S}_{\text{hist}} = 2.241$. With EPTM we obtain from a Gaussian entropy estimate of $\mathcal{S}_{\text{Gauss}} = 2.221$ (figure 5.4).

5.2 Hard Disk Model

5.2.1 Theory

Here we apply EPTM to a more physical model with a 6-dimensional configurational space, namely a simple hard disk model (figure 5.5) describing approximately the be-

	r_{NN}	$s = 2^{-j}$	S_{Gauss}	$S_{\mathrm{Gauss}} - S$
$\mathbf{X}_0$	0.810×10^{-2}	80×10^{-2}	2.565	4.21×10^{-1}
$\mathbf{X}_1$	0.822×10^{-2}	80×10^{-2}	2.271	1.26×10^{-1}
$\mathbf{X}_2$	0.832×10^{-2}	80×10^{-2}	2.235	0.91×10^{-1}
$\mathbf{X}_3$	0.827×10^{-2}	80×10^{-2}	2.229	0.84×10^{-1}
$\mathbf{X}_4$	0.824×10^{-2}	80×10^{-2}	2.228	0.83×10^{-1}
$\mathbf{X}_5$	0.810×10^{-2}	80×10^{-2}	2.221	0.76×10^{-1}

Table 5.2: Entropy and mean nearest neighbor distances of transformed trajectories. S is the real entropy and S_{Gauss} the entropy estimate using a Gaussian. s is the length scale of the used entropy-preserving transformations.

havior of a Lennard-Jones fluid. For simplicity the two-dimensional system consists of three hard disks with diameter d_s, we refer to as solvent, moving freely in a square box of length a. An immobile disk of diameter d_p is in the center of the square box, exemplarily considered as protein. The interaction-potential between two hard disks is described by

$$V_{\mathrm{HD}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \begin{cases} \infty, & \|\mathbf{x}_i - \mathbf{x}_j\| < \frac{d_i + d_j}{2} \\ 0, & \text{otherwise} \end{cases} \tag{5.4}$$

where d_i and d_j are the diameters of the two interacting disks. The center of mass positions are denoted by $\mathbf{x}_i$ and $\mathbf{x}_j$. The potential function of the considered system is

$$V(\mathbf{x}) = \sum_{i,j=1(i>j)}^{3} V_{\mathrm{HD}}(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^{3} V_{\mathrm{HD}}(\mathbf{x}_i, \mathbf{y}_p) + V_{\Omega}(\mathbf{x}) \tag{5.5}$$

where $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ is the configurational vector of 3 solvent atoms. The protein is fixed at position $\mathbf{y}_p = a/2(1, 1)$, where a is the length of the square box.

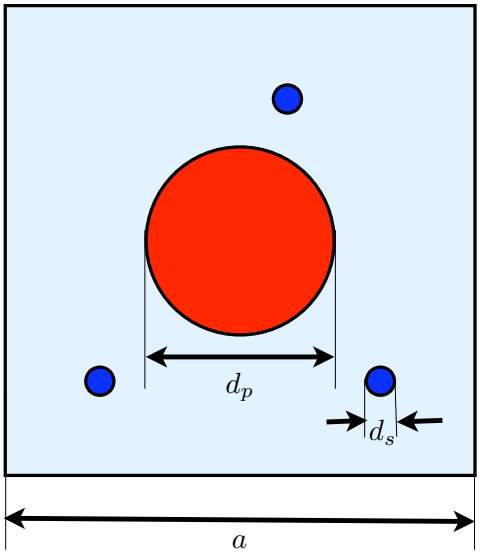


Figure 5.5: Simple hard disk model to simulate the behavior of a solvent-protein system. The model system consist of three solvent atoms (blue) modeled by hard disks of diameter d_s and a protein (red), modeled by a disk of diameter d_p fixed in the middle of a square of length a.

The last term in equation 5.5, a square potential V_Ω

$$V_\Omega(\mathbf{x}) = \begin{cases} 0 & \mathbf{x} \in \Omega \\ \infty & \text{otherwise} \end{cases}, \tag{5.6}$$

was added to constrain the solvent to the square $[0, a]^2$. $\Omega = [0, a]^6$ is the configurational space. Thus, the configurational density is

$$\rho(\mathbf{x}) = \begin{cases} \frac{1}{\xi|\Omega|} & \text{if } V(\mathbf{x}) = 0 \\ 0 & \text{if } V(\mathbf{x}) = \infty \end{cases}, \tag{5.7}$$

where ξ denotes the fraction of configurational space where the potential (equation 5.5)

vanishes. The size of configurational space is $|\Omega| = a^6$. The entropy thus becomes

$$\mathcal{S}[\rho] = k_B(\ln \xi + \ln |\Omega|). \tag{5.8}$$

5.2.2 Simulation

Using the Monte-Carlo Method, we generated several ensembles, each consisting of $30,000$ random configurations drawn from the configuration density 5.7, with diameters of the solvent disks ranging from $d_s = 0$ to $d_s = 3$ in steps of $\Delta d_s = 0.5$. We denote these ensembles by $\mathbf{X}^{(d_s)}$. Permuting the center of masses of the disks in system potential function (equation 5.5) yields the same value, hence, the configurational density is permutation invariant. Therefore, we applied Reinhard's permutation reduction algorithm (PR) to all these ensembles and obtained relabeled ensembles $\mathbf{Y}^{(d_s)}$. Finally, we deformed the relabeled ensembles $\mathbf{Y}^{(d_s)}$ with entropy-preserving maps to $\mathbf{Z}^{(d_s)}$ such that the corresponding densities are more Gaussian (EPTM).

$$\mathbf{X}^{(d_s)} \xrightarrow{\text{PR}} \mathbf{Y}^{(d_s)} \xrightarrow{\text{EPTM}} \mathbf{Z}^{(d_s)} \tag{5.9}$$

For different d_s we determined a nearly exact value of the configurational entropy 5.8 by approximating the ξ with the fraction of accepted configurations during a Monte-Carlo simulation. We denote the the approximation of the exact entropy by $\mathcal{S}$. To compare our newly develop method with Reinhard's permutation reduction, we estimated the entropy of the ensembles $\mathbf{X}^{(d_s)}$, $\mathbf{Y}^{(d_s)}$ and $\mathbf{Z}^{(d_s)}$ using Karplus' formula (equation 3.5), we refer to as Gaussian entropy estimate $\mathcal{S}_{\text{Gauss}}$.

5.2.3 Result

In figure 5.6 we displayed the different entropy estimates. The monte carlo method (red line) provides nearly exact values and reveals the trend that with increasing diameter

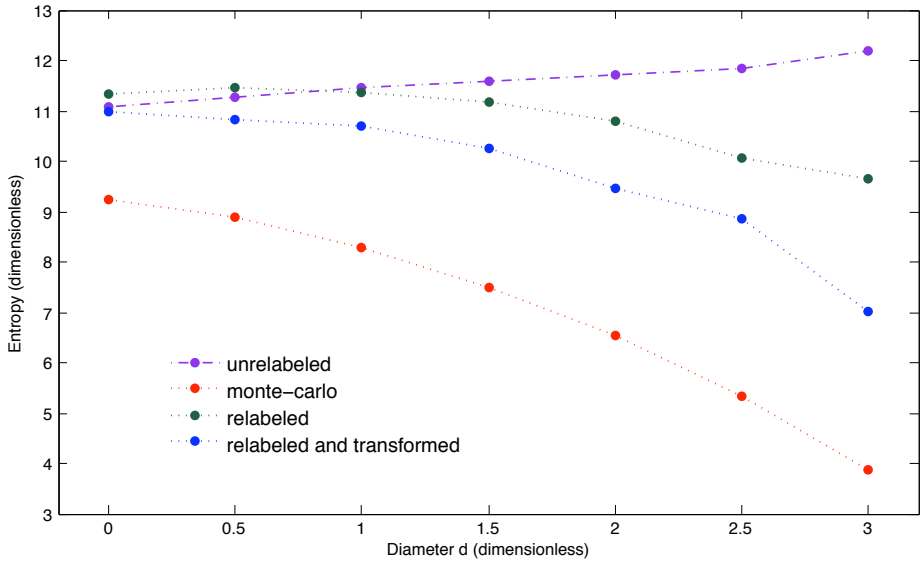


Figure 5.6: Entropy estimates for different diameters d_s of the solvent disks.

d_s of the disk the entropy drops down. This is the behavior we expect, an increasing diameter excludes more volume and, hence, lower the volume of the accessible configurational space. Likewise and increasing diameter of the interacting disks renders the resulting configurational density more anharmonic. A simple Gaussian estimation of the entropy (purple) fails to qualitatively reproduce the trend of decreasing entropy with increasing disk diameter. The Gaussian entropy estimate performed poorly and increases slowly with increasing diameter. Whereas the Gaussian entropy estimate of the relabeled ensemble (green line) reproduces the right trend but fails to give exact values. When transforming the relabeled ensemble via EPTM (blue line) we can improve the Gaussian entropy estimate and preserve the trend however also fail to give exact results (table 5.3).

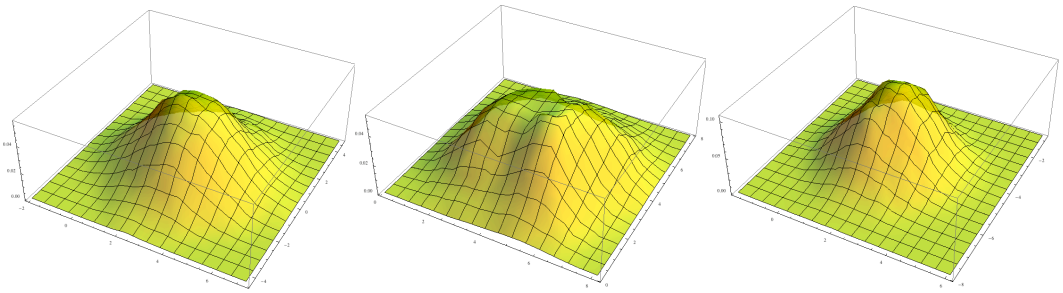
d_s	S	$S_{\text{Gauss}}(\mathbf{X})$	$S_{\text{Gauss}}(\mathbf{Y})$	$S_{\text{Gauss}}(\mathbf{Z})$
0.0	9.24786	11.0736	11.3445	10.9753
0.5	8.89591	11.261	11.4562	10.8339
1.0	8.27552	11.4619	11.3641	10.6939
1.5	7.50074	11.5916	11.1685	10.2675
2.0	6.5323	11.6998	10.7867	9.47263
2.5	5.34661	11.8492	10.0498	8.84414
3.0	3.85853	12.1846	9.6511	7.00621

Table 5.3: S is the approximation of the exact entropy value of the hard disk system where the solvent consists of disks with diameter d_s. S_{Gauss} is the entropy estimate using a Gaussian approximation of the density. $\mathbf{X}$ is the initial density, $\mathbf{Y}$ the relabeled density and $\mathbf{Z}$ was obtained by applying EPTM to $\mathbf{Y}$ to render it more Gaussian.

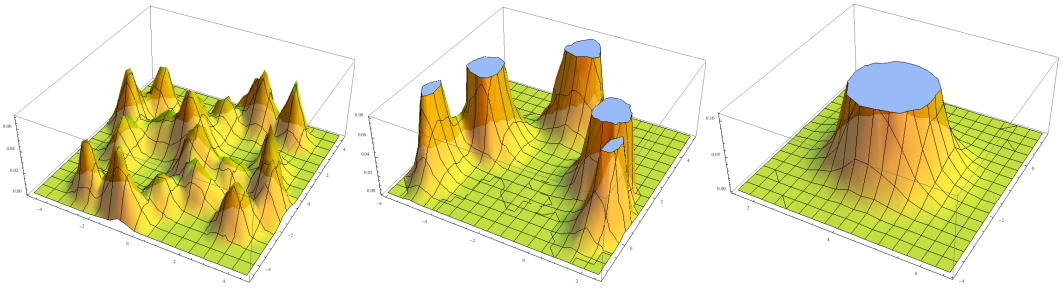
5.3 Discussion

When we consider the projection of the ensemble of the hard disk system on the first two eigenvector derived from its covariance matrix (figure 5.7), we see that for $d_s = 3$ the projected densities of $\mathbf{X}$ and $\mathbf{Y}$ are quite unharmonic. Therefore, the Gaussian entropy estimate will overestimate the real entropy. Applying EPTM to $\mathbf{Y}$ renders the density more Gaussian ((c) and (f)) and improves the Gaussian entropy estimate. However, EPTM fails to extract the analytic value.

So the question arises, how closed can we get to the analytic value? The result from the simple two dimensional densities suggests that we can only get closed to an estimate value that we might obtain from other estimation methods. Obviously, the more points we have, the better we can reproduce the analytic value.



(a) $\mathbf{X}$, $d_s = 0$ (b) $\mathbf{Y}$, $d_s = 0$ (c) $\mathbf{Z}$, $d_s = 0$



(d) $\mathbf{X}$, $d_s = 3$ (e) $\mathbf{Y}$, $d_s = 3$ (f) $\mathbf{Z}$, $d_s = 3$

Figure 5.7: We consider a system of three hard disks with diameter d_s trapped in a square box with an immobile hard disk with diameter $d_p = 2$ in the center (figure 5.5). From the Boltzmann distribution we draw trajectories $\mathbf{X}$ with disk diameters ranging from $d_s = 0, ..., 3$. We applied Reinhard's relabeling algorithm to the trajectories and obtain the relabeled trajectory $\mathbf{Y}$. With an entropy-preserving map we transformed $\mathbf{Y}$ to $\mathbf{Z}$ such that it becomes more Gaussian. In figure (a), (b) and (c) we depict the densities of $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ for $d_s = 0$ projected on the first two eigenvectors from their covariance matrices. In figure (d), (e) and (f) we show the the projected densities projections for the case $d_s = 3$.

6 Summery and Conclusion

In Molecular Dynamics simulations of a solvent-protein system the solvent and the protein have both different dynamical behavior. Proteins behave quite harmonic, since they fluctuate in the vicinity of a well-defined structure of the protein. Hence, the protein is constrained to a small part of its configurational space. With Schlitter's formula we have a straightforward method to estimate the total entropy of proteins using the covariance matrix. In contrast, when treating solvents we encounter two major problems.

First, the solvent density exhibits a complex analytic structure that renders it infeasible to compute integral expressions involving the density such as the entropy. The complex structure is due to the repulsive potential between overlapping molecules, that give rise to parts of the configurational space that is inaccessible, hence, creating holes in the density distribution. Second, the configurational space of the solvent can be sampled only poorly because of the diffusive motion of the solvent molecules.

Tackling the sampling problem was approached by F. Reinhard. He developed a transformation exploiting the permutation symmetry of the solvent. He showed that the trajectory can be projected into a reduced space and the corresponding density can be compressed by N!, where N is the number of solvent particles. Motivated by the fact that the trajectory is more localized, he estimated the translational entropy for different examples using a Gaussian approximation. However, he found out that estimating the entropy with a Gaussian is not the best choice to produce accurate

entropy estimates of a relabeled density. By a simple model we demonstrated that relabeled a trajectory may still be too anharmonic to be fitted by a Gaussian.

In a new approach we tried to improve the Gaussian entropy estimate of the relabeled trajectory using entropy-preserving transformations warping the holes away and rendering the trajectory more Gaussian. To this end, we first developed a theoretical framework that enabled us to easily construct arbitrary smooth entropy-preserving transformations. Choosing the entropy-preserving transformation that could deform the trajectory into a more harmonic one, turned out to be a minimization problem in the space of entropy-preserving maps. We developed an iterative minimization algorithm in a C program that finds the "optimal" entropy-preserving transformation for a given trajectory and transform it into a more Gaussian trajectory. From the Gaussian trajectory we can estimate the entropy simply using a Gaussian approximation of the density.

As prove of principle, we finally applied our newly developed entropy-preserving transformation method (EPTM) to three ensembles. The first two two-dimensional ensembles whose densities feature holes at the surface o the interior served to demonstrate that EPTM is able to find entropy preserving transformations that remove holes and warp the density to a circular Gaussian distribution, whose entropy can be easily estimated. As a more physical example we mimicked a protein-solvent system by considering a simple hard-disk system consisting of three solvent resulting in a 6-dimensional configurational space. Applying EPTM to relabeled ensembles we could improved the Gaussian entropy estimate preserving an important trend of the system, entropy decreases with increasing coupling between the solvent.

EPTM in combination with Reinhard's permutation reduction allows to determine an upper bound of the entropy that is in case of an unharmonic system significantly lower than an Gaussian entropy estimate. Furthermore, EPTM can not only be used to determine solvent entropies but also can serve as a method to improve entropy

estimates of proteins which exhibit in a few cases an high dimensional unharmonic density (reference, ask ulf).

7 Outlook

The algorithm present in the previous work is able to deform a given trajectory in an entropy-preserving manner into a trajectory that is more Gaussian and enables to estimate the entropy using a Gaussian approximation of the corresponding density. However, there are three important theoretical aspects that we have to address to further improve the entropy estimation.

First, we need further analysis on the length scale of the entropy-preserving transformations. A rigorous analytic criterion has to be elaborated how to choose the length scale of entropy-preserving maps for a given trajectory. The situation is similar to the problem of choosing the optimal bin size for a histogram of a data set, whose underlying density we seek to determine. Is the bin length to small we obtain a spiky histogram, is it too large we are smoothing too much and lose information. As for the length scale of entropy-preserving transformations: A too large length scale produces a simple translation or rotation of the trajectory without deforming it. In contrast, a to too small length scale that would compresses the trajectory to a single point, end hence underestimate the entropy.

Second, we have to develop a clear notion what entropy-preserving is or equivalently what means volume-preserving in case of discrete points sets drawn from a density. Intuitively, it seems to be lucid that somehow the nearest neighbor distance must be preserved. Two simple volume-preserving transformations, both translation and rotation preserve the mean nearest neighbor distance.

Third, we need to consider the rotational entropy. A possible approach is using an expansion of orientational correlation functions (reference). In real systems solvent molecules such as water have rotational freedom. With Reinhard's relabeling and EPTM, however, we can only determine the translational entropy of the solvent.

Another aspect is, that EPTM needs to be tested on real systems. As a first simple test,we will apply Reinhard's permutation reduction and EPTM to an argon gas — a Lennard-Jones fluid —, which features the behavior a solvent. We will estimate its configurational entropy for different coupling constants between the argon atoms. Corresponding to our simple disk model (reference) we expect a decrease of entropy with increasing coupling parameter of the Lennard-Jones Potential. In a next step, we will mimic the presence of a protein with dummy a simple soft-core potential in the center of the simulation box.

8 Appendix

8.1 Documentation of g_entropyestimate

g_entropyestimate is called from command line. It reads either a trajectory, binary file or text file produced by an MD Simulation and searches for the entropy-preserving transformation $\mathbf{f_c}(t, .)$ minimizing the upper bound of mutual information as we have described in chapter 4. The input syntax is

```
g_entropyestimate -xtc traj.xtc log_traj.bin t s
```

where t is the flow time and $s = 2^{-j}$ the spatial resolution of the wavelets that are used to construct entropy-preserving transformations. **log_traj.bin** is a binary file containing the wavelet coefficients and according indices (equation 4.41) corresponding to two-dimensional entropy-preserving transformation. Furthermore, it contains the initial and the transformed trajectory as well as entropy estimates.

8.2 Volume preserving maps

We prove that the following identity holds for all continuous maps $\mathbf{f_v}(t, \mathbf{x})$ which constitute the solution at time t of the movement of $\mathbf{x}$ in a velocity field $\mathbf{v}$ (equation 4.10).

$$\partial_t \ln \det \left(\frac{\partial \mathbf{f_v}(t, \mathbf{x})}{\partial \mathbf{x}} \right) = (\nabla \mathbf{v}) \left(t, \mathbf{f_v}(t, \mathbf{x}) \right) \tag{8.1}$$

Using Jacobi's identity for an invertible square matrix $\mathbf{C}$: $d \ln \det(\mathbf{C}) = \operatorname{tr}(\mathbf{C}^{-1} d\mathbf{C})$ the above equation can be proven easily equation

$$\begin{aligned}
\partial_t \ln \det\left(\frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right) &= \operatorname{tr}\left[\left(\frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right)^{-1} \partial_t \frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right] && (8.2)\\
&= \operatorname{tr}\left[\left(\frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right)^{-1} \frac{\partial \partial_t \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right] && (8.3)\\
&= \operatorname{tr}\left[\left(\frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right)^{-1} \frac{\partial \mathbf{v}\left(t, \mathbf{f_v}(t,\mathbf{x})\right)}{\partial \mathbf{x}}\right] && (8.4)\\
&= \operatorname{tr}\left[\left(\frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right)^{-1} \frac{\partial \mathbf{v}}{\partial \mathbf{x}}\left(t, f_{t,v}(x)\right) \frac{\partial \mathbf{f_v}(t,\mathbf{x})}{\partial \mathbf{x}}\right] && (8.5)\\
&= \operatorname{tr}\left[\frac{\partial \mathbf{v}}{\partial \mathbf{x}}\left(t, \mathbf{f_v}(t,\mathbf{x})\right)\right] && (8.6)\\
&= (\nabla \mathbf{v})\left(t, \mathbf{f_v}(t,\mathbf{x})\right) && (8.7)
\end{aligned}$$

8.3 Gradient ODE

We develop the time evolution of the gradient of the entropy-preserving map $\mathbf{f_c}(t,.)$ (definition 4.42) in respect to the wavelet coefficients $\mathbf{c}$. For this end, we introduce the notation $\partial_{\mathbf{l}} \equiv \frac{\partial}{\partial c_{\mathbf{l}}}$, where $\mathbf{l}$ is the wavelet coefficient index vector (equation 4.41). Let $\mathbf{y} = \mathbf{f_c}(t,\mathbf{x})$. The time evolution of the gradient $\partial_l \mathbf{y}$ is

$$\begin{aligned}
\partial_t \partial_{\mathbf{l}} \mathbf{y} &= \partial_l \partial_t \mathbf{y}\\
&= \partial_l \left\{\mathbf{v_c}(\mathbf{y})\right\}\\
&= \left(\partial_l \mathbf{v_c}\right)(\mathbf{y}) + \mathbf{J_{v_c}}(\mathbf{y}) \cdot \partial_l \mathbf{y}\\
&= \boldsymbol{\psi}_{\mathbf{l}}^{\mathrm{div}}(\mathbf{y}) + \mathbf{J_{v_c}}(\mathbf{y}) \cdot \partial_{\mathbf{l}} \mathbf{y} && (8.8)
\end{aligned}$$

where $\mathbf{J}_{\mathbf{f}_c}$ is the Jacobian matrix of $\mathbf{f_c}$. As initial condition we obtain

$$\partial_{\mathbf{l}}\mathbf{y} = \partial_l \mathbf{f_c}(0, = \partial_l \mathbf{x} = 0. \tag{8.9}$$

Bibliography

[1] I. Andricioaei and M. Karplus. On the calculation of entropy from covariance matrices of the atomic fluctuations. , 115:6289–6292, October 2001.

[2] Y. Brenier. TOPICS ON HYDRODYNAMICS AND VOLUME PRESERVING MAPS. *Handbook of Mathematical Fluid Dynamics II.(North-Holland, Amsterdam, 2003)*, pages 55–86.

[3] E. Deriaz and V. Perrier. Towards a divergence-free wavelet method for the simulation of 2D/3D turbulent flows. *Arxiv preprint cs.NA/0502092*, 2005.

[4] E. Deriaz and V. Perrier. Divergence-free and curl-free wavelets in two dimensions and three dimensions: application to turbulent flows. *Journal of Turbulence*, 7(3):1–37, 2006.

[5] P. Echenique. Introduction to protein folding for physicists. *eprint arXiv: 0705.1845*, 2007.

[6] Y. Harano and M. Kinoshita. Translational-Entropy Gain of Solvent upon Protein Folding. *Biophysical Journal*, 89(4):2701–2710, 2005.

[7] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. J. Wiley New York, 2001.

[8] Y. Kanazawa. An Optimal Variable Cell Histogram Based on the Sample Spacings. *The Annals of Statistics*, 20(1):291–304, 1992.

[9] M. Karplus and J.N. Kushick. Method for estimating the configurational entropy of macromolecules. *Macromolecules*, 14(2):325–332, 1981.

[10] K.H. Knuth. Optimal Data-Based Binning for Histograms. *Arxiv preprint physics/0605197*, 2006.

[11] R. König and T. Dandekar. Solvent entropy-driven searching for protein modeling examined and tested in simplified models. *Protein Engineering Design and Selection*, 14(5):329–335, 2001.

[12] P.G. Lemarie-Rieusset. Analyses multi-resolutions non orthogonales, commutation entre projecteurs et derivation et ondelettes vecteursa divergence nulle. *Rev. Mat. Iberoamericana*, 8(2):221–237, 1992.

[13] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

[14] L. Parra, G. Deco, and S. Miesbach. Redundancy reduction with information-preserving nonlinear maps. *Network: Computation in Neural Systems*, 6(1):61–72, 1995.

[15] A. Patrascioiu. THE ERGODIC-HYPOTHESIS. *Los Alamos Science Special Issue*, 1987.

[16] W.H. Press, S.A. Teukolsky, and W.T. Vetterling. *Numerical recipes in C++: the art of scientific computing*. Cambridge University Press.

[17] F. Reif and W. Muschik. *Statistische Physik und Theorie der Wärme*. de Gruyter, 1987.

[18] F. Reinhard and H. Grubmüller. Estimation of absolute solvent and solvation shell entropies via permutation reduction. *The Journal of Chemical Physics*, 126:014102, 2007.

[19] H. Schäfer, A.E. Mark, and W.F. van Gunsteren. Absolute entropies from molecular dynamics simulation trajectories. *The Journal of Chemical Physics*, 113:7809, 2000.

[20] J. Schlitter. Estimation of absolute and relative entropies of macromolecules using the covariance matrix. *Chemical physics letters*, 215(6):617–621, 1993.

[21] A. Tveito and R. Winther. *Introduction to Partial Differential Equations: A Computational Approach*. Springer, 1998.

[22] K. Urban. *A Wavelet Galerkin Algorithm for the Driven Cavity Stokes Problem in Two Space Dimensions*. Inst. für Geometrie und Praktische Mathematik, 1994.

[23] K. Urban. On divergence-free wavelets. *Advances in Computational Mathematics*, 4(1):51–81, 1995.

[24] A. Wehrl. General properties of entropy. *Reviews of Modern Physics*, 50(2):221–260, 1978.